

SyncWare News

The journal for technical
applications of T/S
computers

Volume 5 Number 6

Final Edition



SWN MERGER ANNOUNCED

Our sincere apologies for the extreme delay in getting this issue (and the previous one) to you. We didn't intend it this way, but the delay was a prelude to even sadder news.

This is the last issue of SyncWare News that will be issued under that name. With this issue, we complete 5 years. We believe that we have performed a service for users of the Sinclair line of computers, and we believe that our readers have appreciated that service. For our part, we certainly appreciate your loyalty, your help, and your encouragement. However, we just can't make it any longer.

We thought that we had an arrangement whereby another magazine of comparable quality would provide some continuity for our readers. And for our authors. Unfortunately, the deal fell through--without rancor on either side, we hope. But we still don't want to leave you in complete bereavement.

So we propose that SyncWare News be absorbed by our sister publication, Quantum Levels. For those who agree, we will enter or extend your subscription to Quantum Levels on a copy-for-copy basis. And each issue of Quantum Levels will include several pages of material relating to the good old ZX81/TS1000 and TS2068. For each of you who does not want to continue on this basis, and still has an unexpired subscription, we will return the money representing your unfilled subscription.

Send an indication of your wishes to our publisher,

Jeff Moore
602 South Mill Street
Louisville, Ohio 44641

See FORUM article "From The Publisher's Desk.." for further details.

For Your Support

ZEBRA SYSTEMS, INC., 78-06 Jamaica Ave., Woodhaven, NY 11421, (718) 296-2385 is currently offering Graphics Collection tapes for use with their TS2068 Graphics Designer Series programs: The Banner Designer, The Greeting Card Designer, and the Sign Designer. Each tape includes 30 graphics. Collection subjects include holidays, office, sports, and religion to name but a few. There are currently 12 collections available with more under development. Contact Zebra for details and price.

THE CAPITAL AREA TIMEX SINCLAIR Users' Group will be sponsoring the C.A.T.S. CAPITAL FEST, May 5-7, 1989. The fest will be held at the Route 450 and Beltway Howard Johnson's in New Carrollton, MD. For more information regarding the event of 1989 write to: C.A.T.S. Capital Fest, P. O. Box 24, Garrett Park, MD 20896-0024. SyncWare News will publish more details as they are made available to us.

SINCLAIR USERS UNITE!! Sign up for SNUG! A Sinclair Northamerican Users' Group is being formed. It will provide all of us with a unified voice to vendors, the

latest product info, a list of other members, and the largest library of Public Domain and Shareware in the United States. For more information write: SNUG, 7515 Arbordale Dr., Port Richey, FL 34668.

SINCUS NEWS is the newsletter of the Sinclair Computer Users' Society, a non-profit organization operated by volunteers dedicated to the Sinclair and Timex/Sinclair computer user. \$8 for six issues. Write: Paul Hill, SINCUS, 1229 Rhodes Rd., Johnson City, NY 13790.

SyncWare News

Jeffrey D. Moore
Basil Wentworth
Fred A. Nachbaur
Thomas J. Bent

Publisher
Editor
Technical Director
Assistant Editor

All subscription information, billing, and ad artwork should be addressed to Jeff at:

SyncWare News
602 South Mill Street
Louisville, OH 44641
(216) 875-1257

Article submissions, Forum, Support listings, Classified ads, and really easy questions should be addressed to Basil at:

SyncWare News
1413 Elliston Drive
Bloomington, IN 47401
(812) 336-0837

Real tough questions, matters concerning hardware, and everything else should go to Fred at:

SyncWare News
C-12, Mtn. Sta. Group Box
Nelson, BC V1L 5P1
CANADA
(604) 354-3858

Back issues of SWN are available for \$3.50 each. Vol. #1 (for the 1000) is available for \$16.95, US postage included.

SWN is done entirely on TS computers. Submissions are preferred as word processed text files recorded on good quality tape. Memotext for the 1000 and Mscript for the 2068 are the preferred word processors. Documents submitted in Tasword Two or other word processors must be done in 52-column format. If you do not have a word processor, but you feel you have a good idea, then don't hesitate to work it up and submit it. Complete writers' guidelines are available upon request from the Indiana editorial office. Advertisers' rate cards from the publisher, at the Ohio address.

SWN pays authors about \$10 per page for original articles. Payment is made at time of publication.

AFR SOFTWARE (R)

Presents:
Powerful And Inexpensive
Business Software
For "Timex-Sinclair"
Computers

WORD PROCESSING

T/S-TEXT 2000\$19.95
ZX-TEXT\$19.95

SPREADSHEET CALCULATOR

T/S-CALC 2000\$19.95
ZX-CALC\$19.95

CYCLE ACCOUNTING

T/S-ZX Financial Report Generator\$29.95
Printout Of Same\$13.00

APPOINTMENT SCHEDULER

T/S-CALENDAR 2000\$19.95
ZX-CALENDAR\$19.95

Send S.A.S.E. For Free Catalog
Or Check Or Money Order To:
A.F.R. SOFTWARE
1605 Pennsylvania Ave.
No. 204
Miami Beach, FL 33139
(305) 531-6464
FLORIDIANS ADD SALES TAX
Dealer Inquiries Invited

Forum



From the Publisher's Desk...

As many of you may already know, SyncWare News is published by four partners who are geographically spread out across continental North America. All of us, with the exception of Basil who is retired but very active in the Symphony, college, and civic matters, work full time jobs to support our families. Due to the cooling of the Timex/Sinclair market, we have been publishing more or less as a hobby.

For the past two years or so, SyncWare News was taking in enough to pay for its authors, printing, and postage. We have done everything we could think of to cut the magazine's expenses. Authors have been asked to take a cut in pay. The magazine is being mailed 3rd class. Each of the partners has been paying his own incidental office expenses (telephone, blue pencils, tape, etc.) out-of-pocket for some time. Nevertheless, after the publication of the Volume 5 Number 5 issue of SyncWare News, we found ourselves on the brink of bankruptcy. There was no longer enough income to cover the major expenses for SyncWare News. At this point, we were forced to consider the probability of closing down the magazine. In an effort to gain some advice on how to gracefully go out of business, we explained our situation to a few close friends.

During July and August, we received not one, but two offers from parties interested in acquiring SyncWare News. September and the better part of October found us in negotiations with these parties to sell out SyncWare News. (During this negotiation period, we held, rather than return, many of the checks we received for renewals to SyncWare News. We felt at the time this was best due to the uncertainty of the final outcome of the negotiations. If SyncWare News was sold, the checks would be cashed and the subscriptions continued as normal. If not, the checks could be returned for only the cost of postage, saving both the subscriber and us the cost of processing the check.) In the end, we felt that neither offer would be fair to us or the current subscribers to SyncWare News. In late October we decided to continue with SyncWare News, but to change its format somewhat. We cashed the renewal checks we'd been holding and then began work on SyncWare News Volume 5 Number 6.

Beginning with Volume 6 Number 1, SyncWare News will merge with its sister

publication Quantum_Levels. If you are a current subscriber to SyncWare News rest assured, you will receive every issue due you. We will add your name to the Quantum_Levels mailing list and continue your subscription with the new combined magazine. If you are not satisfied with the new format after receiving your first copy or two, we will be more than happy to refund the balance of your subscription.

The merger will allow us to be more innovative and explore some exciting new areas. We will continue to run quality articles on TS1000 and TS2068 software and hardware. The QL computer will be explored in depth. The new Cambridge Computer Z88 (Sir Clive's laptop) has many interesting features and deserves to be supported as well. Also recently announced and in distribution is the Sinclair MS-DOS computer (an Amstrad clone of an IBM clone!?!). Many of you have requested information for this machine.

So you can see, the world of Sinclair computers is not dead, but it is changing. SyncWare News, together with Quantum_Levels, will continue to adjust accordingly. As we make alterations we will need your input. This publication has always leaned more or less toward the TS 'power user' - those who wished to explore beyond the horizons set forth by an owner/operator's manual. We wish to continue to be that small voice which beckons those who dare to push their minds and machines to the performance envelope and beyond. Help us serve you. Send us your ideas, thoughts, questions, and solutions.

We thank you for your continued support.

Sincerely,
Jeffrey D. Moore, Publisher
SyncWare News
Quantum_Levels

LARKEN 3 1/2" DISK SYSTEM for \$199.95

Package includes ...

- Larken 2068/Spectrum Disk Interface and LKdos Cartridge
- 1 - 400K 3.5" Mitsubishi Disk Drive and Power supply
- or 1 - Single sided 5 1/4" Disk drive and Power supply
- 1 Disk of Disk converted programs to start you off.
- Other Drive combinations available.
- Ready to Run Floppy disk System . 90 day guarantee
- Limited Time Offer, Until 3.5" 's are gone .

(prices are US \$ add \$10 S&H)
LARKEN ELECTRONICS, RR#2 NAPAN ONTARIO, K4B-1H9 CANADA
(613)-835-2680

Random Access From Our Readers

One of our readers (whom I won't identify) wrote to point out that I had omitted the letter "u" from "florescent" in one of our recent issues. Seemed to think that I might have left it out on purpose, to avoid running the risk of the common misspelling, "flourescent". Sorry about that, but they're two different words. I meant "florescent", as contrasted with "budding". But it is tempting to expect that the familiar word was intended when you encounter a seeming aberration like "cryptogam" or "Calvary". Or, as Leslie Charteris pointed out, "formication".

* * * * *

To Warren Fricke, of Depew, NY: Thanks for your offer of help, and for the fine words you had for SyncWare News. You have been a true friend. I'm sorry that we aren't able to take advantage of your offer this time.

* * * * *

TS2068 TO TTL VIDEO

My question is on using a monitor like the one you can get from BG Micro. How would you construct an interface to drive one? I'm talking about the type of TTL CRT that would require you to take the NTSC signal from the 2068 and split it out to drive this 'monitor'. I actually know too little of what I'm talking about and so the question.

John J. Shephard, III
Coldwater, MS 38618

How about it? Can anyone out there help John?

The Shows

I had the good luck this summer to attend both of the big Summer shows--Portland and Cleveland. Both were well presented, and gave me a chance to pick up some bargains, as well as to talk with old (and new) friends. Attendance at each show was reported at about 150, and each show was complemented by a series of seminars led by the big names in the field.

By far the stellar attraction at both shows was Nigel Searle. Nigel has been associated with Sir Clive Sinclair for many years, and has a wealth of anecdotes that kept the audience both informed and entertained. Nigel is a virtuoso speaker--he could make an adventure out of reading a circuit diagram. If you ever have a chance to hear him, RUN--don't walk.

S.N.U.G.

Mel Nathanson made a telling presentation for SNUG (Sinclair National--or North American--Users' Group) at the Cleveland show, provoking a lot of serious discussion. Clearly, SNUG is now in about the same position that our Constitution was in when the Continental Congress was convened. And much of the debate centered on the same question--in those days, whether the new Union would represent the states or the people. In the SNUG context, the controversy centers on whether membership will be open to UG's or individuals, or both. My own suggestion is that SNUG should be redefined as Syndicate of North American Users' Groups, but that affiliation should be available to individuals as well. (Or could you regard an individual as a User's Group with a membership of one?)

At any rate, what is needed (I think) is a re-enactment of that Continental Congress I mentioned above. It doesn't have to be an actual assembly of persons--in these days, communications permit a meeting of the minds by remote control. Maybe someone will volunteer to be their Thomas Jefferson, and submit a set of proposals that the assembly can discuss. Maybe there'll be several such volunteers.

In the meantime, SNUG needs some money to get things going. Mel has suggested an interim annual dues structure of \$12 per person, and \$15 per group. Make your checks out to SNUG. I have already given him mine. Mel has promised that he'll return all money (I'd forgive him if he deducted the cost of returning the check) if the project doesn't appear to be off the ground by New Year's. His idea of off the ground is 100 members by the first of January.

Send checks and ideas to

SNUG
c/o Mel Nathanson
7515 Arbordale Drive
Port Richey, FL 34668
tel (813) 863-5552

TS1500 Mod for Monitor

Dick F. Wagner
Canby, OR 97013

(Editors Note: SyncWare News was the first to publish a circuit for adapting the TS1000/TS1550 to a monitor back in 1983. See SyncWare News Volume 1, page 61 thru 63 for our in-depth circuit description. We are publishing Mr. Wagner's circuit because it is not only TS1500 specific, but has a very well done parts layout drawing. This excellent drawing will make this mod easy for even the novice hardware hacker. - ed.)

In answer to Gerard Tripptree's request for information to modify a TS 1500 to drive a composite monitor, there is a simple and easy hardware solution. Time Designs magazine, Sept-Oct,

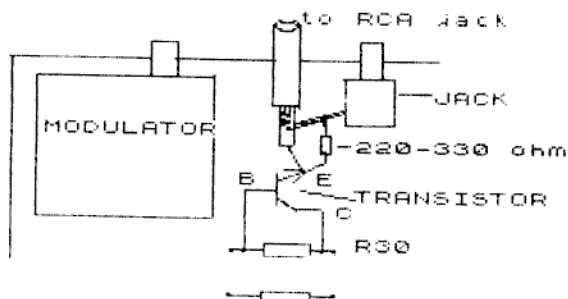
1985 published an article I wrote on the subject so interested readers could go to this back issue for details.

However, here is enough information for readers with some expertise with a soldering iron and electronics to do the job. The method is that used to connect the computer signal of a TS 1000 to a monitor by using an emitter follower amplifier.

The diagram shows the simple amplifier used--a 2N2222 transistor (Radio Shack 276-2009), and a 220-330 ohm one quarter watt resistor. I assembled this by bending one resistor lead close at right angles and soldering it to the transistor emitter lead close to the bottom of the transistor and so the resistor is tight to the flat of the transistor. Clip the resistor wire close to the emitter lead. This is the amplifier that receives the signal on its base, +5 volts on the collector, and output from the emitter lead with the resistor terminated to ground. Suitably insulate with plastic tape, shrink tubing, etc. This amplifier works on composite signal monitors designed for 75 ohm input impedance.

Computer resistor R30 is the signal source for the TV modulator. This resistor is near the lower right corner of the modulator box on the circuit board component side. The left end nearest the box is the signal end and the right end is +5 volts. Connect the amplifier to the two leads of R30 as per the diagram. Use a low heat soldering iron for all soldering, and of course only rosin solder.

Various methods can be used to connect the amplifier emitter to the monitor. I prefer to use a 6-8 inch shielded lead with a molded RCA phono jack such as RS 274-337 cable so the regular TS TV cable



TS 1500 Monitor Adapter

can be used. The center lead is connected to the transistor emitter. The woven shield is soldered to the ground terminal of the near jack as shown. This is also a good strain relief.

The shielded cable will exit from the rear of the case. File a suitable notch in the case edge between the jack and the TV modulator box to fit the cable. Be very careful in replacing the keyboard ribbon cables. Long nose pliers may be helpful in this task.

RMG Enterprises will do this conversion for readers not wishing to do the work themselves. Ship your TS 1500 computer (no power supply) to 1419-1/2 7th St., Oregon City, OR 97045 and include \$14.95 + \$3.00 P & H. Phone is 503-655-7484. (Note: when you make a phone call, remember that Oregon is in the

Pacific Time zone--Ed.) They will quickly return your computer with conversion, including a 6-8 inch long cable with RCA type jack connection.

TYD☆BYTS

SICK TRANSIT . . .

Probably most of you have received notice of the bankruptcy of QUANTUM COMPUTING. Apparently, the notices were sent to everyone on the company's customer list. I doubt that there is much hope of collecting anything from the company, as the notice listed unsecured debts as over \$60,000, with total assets of \$0.01 (that's one cent). I don't know just how one can measure assets that small, but suspect that there is a bit of a rounding error involved.

However, for anyone who wants the information, the appropriate names and addresses are:

Frank Tomel, jr.
119 Schmitz Terrace
Mt. Arlington, NJ 07856-1208

Interim Trustee: Jonathan Kohn
1180 Raymond Blvd.
Newark, NJ 07102-4107

Attorney for the Debtor: Robert G. Lavitt
Wiley, Malehorn & Sirota
250 Madison Ave, Drawer 210C
Morristown, NJ 07960

.....
The recent issue of Radio-Electronics magazine announces a lithium battery of AA size, manufactured by Eveready. The battery is described as having "unmatched energy density". It is said to last up to double the length of alkaline batteries "in many applications", and to have a shelf life of "10 years or more".

The batteries are expected to be on the market in early 1989. Z88 users, rejoice!

.....
Jocelyn is a member of a Home Economics club sponsored by the County Agent of the U. S. Department of Agriculture (USDA). The latest bulletin from the USDA's local office warns about the danger that your home-owner's insurance policy may not cover your computer.

It seems that many policies do not cover losses to equipment that is used in business. And some insurance companies are almost as exacting the Olympics committee in their definition of amateur status--you lose your "non-business" standing as soon as you earn ANY money, however little, with your computer.

And then there is the question of what risks are covered. Maybe you're ok as far as fire and theft go, but what if your damage results from voltage surges, or immersion in pea soup, or a fall from the desk onto that nice new slate floor? And does your policy cover only the present value of your computer (which may be pretty low), or its replacement value (which may be considerable)?

The difference in cost between skimpy and adequate coverage may be ridiculously low. Only your insurance agent knows for sure, as the old hair-color ad used to go. Ask him.

NVM and the 2068

Paul Holmgren

Often while folks gather and talk about their computers I hear something like this, "Why don't you try this" and the almost standard reply is "I didn't know you could do it that way, thanks" This article is the result of my being in that same boat and finding an answer by myself.

Way back when, really just in May 1987 during the T/S Midwest Computer Show held in Indy, I found time to visit with Tom Bent and Tom Woods. The purpose was to get one of the 64K Non-Volatile RAM (NVM) boards they were selling for the Timex 2068. The current version uses 43256 120ns type 32K static RAM chips allowing you to operate with 32K or 64K of RAM on the board and retain the 43256 chip memory with a battery backup method. Current battery life expectancy reaches beyond the life of the 2068 (or of you and me) if you follow the guide lines in the owners' manual.

If only I could . . .

One of my major problems in writing a BBS program in BASIC is/was, the size of the program directly affects the amount of features and messages it can contain in 38K of HOME RAM in an unexpanded 2068.

I wanted to experiment with the 64K unit by placing the BBS program in the Timex Command Cartridge port. I wanted to escape the problems of trying to fit additional features in the BBS program and retain the original size of the message base in the computer's memory. I was excited with the prospects of the additional features I could add to the program with an extra 32K or 64K of RAM available, and still retain the original message base size the program uses.

After the dust from handling the Show settled I placed an order with my local mail-order house for 2 43256 32K static RAM chips and read/reread the manual for the board until the chips arrived.

Well, to put it mildly, I failed to understand everything in the owner's manual, even after 2 or 3 readings. I followed the directions as I understood them. Every time the program I installed in the NVM memory ran and tried to use the short section of machine code it crashed on me. Sure, there was something I was doing wrong, but what, I didn't realize.

So I set the darn thing aside and proceeded onto other things. Like learning to live with my Oliger Disk interface. Even the Wife became interested in the 2068 after the disk setup was finished.

Fits and flashes in the night . . .

In Sept. 1987, our user group received about 2 MEGA bytes of public domain programs for the 2068. The first thing we did was to place all those programs on disk. In an effort to clean up, weed out duplicates, and to reduce the total disks needed, I had to come up with a way of managing my disks. I ended up with a program that did what I wanted, but every time I had to FORMAT a disk I had to LOAD the file 0 loader program that Oliger's SAVE program

can use, SAVE it to file 0, then reload my Disk File Manager program. I needed a better way.

DING!!! a light went on. I have this NVM board sitting there doing NOTHING. Sure would be nice to put my manager program on it so that I could use it without having to do a lot of disk swapping. In a fit of late night brilliance everything finally clicked. I had made it! My programming efforts were now using the 64K NVM board in the DOCK port and machine code worked, DATA/READ worked, operating the Oliger disk IF from basic in the NVM board worked. Now I can use the DOCK port memory and have the original 38K of HOME bank memory available also. At this time I am using the NVM board to contain a program and the original HOME bank of memory holds all the variables used by that program while ANOTHER BASIC program is loaded. So I have one program in the DOCK port, another program in the 2068's HOME memory, and plenty of room for variables.

Comes the dawn . . .

So far, this is what I have learned, and most importantly written down while the insight was fresh. More may follow as I become more comfortable with all this.

If any program you wish to place on the NVM board contains machine code, it should be programmed to operate and be stored starting at location 26688. (There is another way to run machine code out of the DOCK port, but that secret escapes me so far.) Then include a subroutine in your main program to POKE the code into memory starting at 26688. Make sure you POKE all your code. I used a DATA/READ method, and it worked with raw numbers and with a string variable. That should be relatively easy to do. (I know, I know--famous last words). Also try to make sure all the bugs are dead in the rest of the your program.

Next VERY carefully follow the guide lines in the NVM owner's manual for relocating the 2068 system variable PROG, then the movement of the program you want placed into the NVM board.

NOW for the POKES as given in the manual.

POKE 32768,1 tells the system you have BASIC, or BASIC and machine code to deal with.

POKE 32769,2 tells the system you are trying to use the AROS feature of the 2068.

POKE 32770,8 is the first of a 2 byte indicator to inform

POKE 32771,128 the 2068 that BASIC in the NVM board now starts at 32776.

POKE 32772,15 specifies the memory CHUNKS you are using to hold the program to be run by the computer. That is the NVM's memory area it occupies for your use at this time. (Other uses and a better understanding will allow you to specify use other CHUNKs as needed.)

POKE 32773,1 AUTO-runs the BASIC program contained in the NVM board.

POKE 32774,xx and

POKE 32775,xx the values stored here inform the 2068 of the size of area you need for your machine code. This info is needed so that the 2068 can find

out where the BASIC and variables sections will end up being found in HOME memory. As I understand it, this info changes as you move from DOCK to HOME memory and back. Use the following conventions to determine the values for these 2 POKEs.

32774 will need the value found by $\text{code size} - (\text{INT}(\text{code size}/256) * 256)$

32775 will need this value: $\text{INT}(\text{code size}/256)$.

Remember that a POKE 23750,0 restores the HOME memory for operation of any program there, and POKE 23750,128 turns the DOCK port back on.

Not to worry, if you do all this and find a bug in your program as you use/test it, this is what I do. I remove the NVM board from the DOCK port, carefully slip a piece of paper under the battery contact, then carefully short out the main 220uf capacitor (the round can next to the switches) for a moment so that the static memory chips fully drain and clean its memory, remove the paper and reinsert the NVM in the 2068, power it up, reload the program I am experimenting with, and fix my bugs. Then do it all over again.

If you know of an easier way, I await your tips.

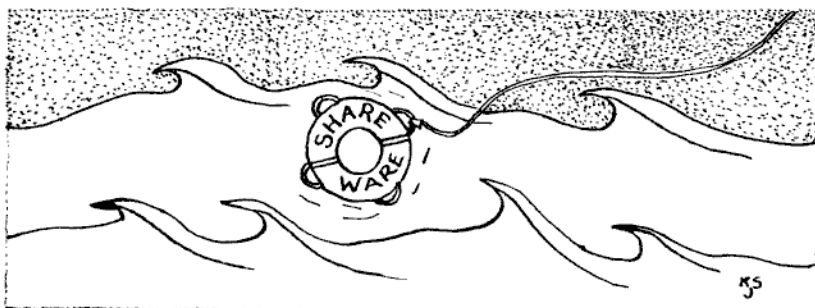
A brighter tomorrow . . .

Right now, as I finish dashing this off, I am reaching for my BBS program and will begin adding the features I wanted as if I had all kinds of memory, which I have now, thanks to the 64K NVM board.

LATE THOUGHT; this is a real neat way to program and test your programs before you place them on an EPROM board to be run out of the DOCK port like the Timex Command cartridges. It sure beats finding a bug in a program and having to erase and then reprogram an EPROM so that your program runs better. I also wonder what this means to those folks toying with the IM2 interrupt experiments. I can see it now: the 2068 with a program in HOME memory, an IM2 utility, and the DOCK port with a program.

MULTITASKING

Can it work? Only time and the talents of fellow Timex/Sinclair survivors will tell.



Share-Ware - - The T/S Lifeline?

F. Nachbaur

(Editor's NOTE: This topic is especially pertinent in view of the plans of SNUG.)

Software has always been a difficult commodity to market and distribute. Users want new software, they want support for their software purchases. What's more, EVERY computer buff likes to build up a big library of programs. Most are never or rarely used, but there is a certain element of the pack rat in all of us.

Suppliers for obsolete computers spend time and effort to produce new products for a dwindling customer base. It has been proven again and again that one stands a slim chance of making a profit in these "low-end" computers. Still, some try. As long as there are buyers, there will be sellers. However, dealers and programmers want their efforts to be financially worthwhile.

Moral Obligation VS Reality

These ideas have often clashed, with the one camp copying software for friends because the program is too expensive, and the other claiming that software is so expensive because many sales are "lost" to illegal clones. This affects not only our tiny corner of the computer world, but all computers and

computerists. A resolution could come in the new software marketing concept called "Shareware" or "Fairware".

Copying programs is a fact of life. You do it. I do it. I've only actually paid for a portion of the programs I have in my collection. It is important to note, however, that out of these programs are a scant half-dozen I use all the time, and maybe a couple dozen I use occasionally. The rest were looked at once or twice, before being enshrined in "the files" (one of two big drawers full of tapes).

The fundamental tenet is, "If you use it, pay for it." The most direct application of the "shareware principle" is to send a "shareware" contribution to the author of a cloned program that you are finding useful. (Obviously, no-one expects you to pay for something that you can't or don't want to use.) Unfortunately, it can be a real chore to locate the actual author of a given program, so all too often we either can't find a program at all, or if we do, we may not know where to send "whatever it's worth". Furthermore, human nature being what it is, it's all too easy to indefinitely procrastinate mailing that contribution.

When we freely clone programs from our friends, we have no recourse if we are missing part of the documentation, or if there are updates or improved versions available. How do you get questions

answered, bugs debugged or at least defanged, gain added technical or background information? How do you reward the creators and developers of the programs you have in your library? Some programs are the product of several diverse individuals; how do you reward each one's part in the product? At the same time, how do you keep your computer from driving you into the poor-house?

The answer to these questions might be to expand and organize the concept of "shareware". I don't know if formal guidelines have ever been published, but "shareware" or "fairware" libraries are simply an organized way of paying fairly for what you are already using, and to make other programs available to you at reasonable cost. Shareware libraries are already in operation all over the continent.

The libraries freely exchange material, or even share a database of available programs. You purchase from your closest library, at a reasonable cost per program. In some cases, users can submit written updates, clarifications, and other discoveries to be included in the documentation file for any given program.

Dubbing, mailing, and otherwise processing orders also needs to be covered and made worthwhile. The library therefore takes a moderate fee for these services, and splits the rest with the authors who contributed.

This is often done quarterly, with payments made in the quarter following the order-taking. A sliding scale can determine how much each author receives; obviously, a massive application should pay better than a simple arcade game. In principle, then, everyone gets a fair share of whatever interest remains in the field of users.

The New ZX World

There is something oddly "different" about the ZX81 family of computers, which spawned a massive wave of interest in the early '80's. Partly because the machine was improperly marketed, partly because development of the TS2068 was completed too late, and largely because of the overall market slump, Timex closed down the Timex Computer division. The TS2068 was in reality never truly completed, and new products that were prototyped never saw the light of day.

Yet in Europe, the ZX Spectrum, the prototype of the TS2068, was the most popular personal computer. (Never mind that it was mostly used for games, and never saw true development of its potentials.) So now we have two quite different machines, one very simple yet powerful, and the other even more powerful, both sharing that unique invention called Sinclair ZX BASIC. Both are proving to retain a lot of interest in a hard core of dedicated fanciers.

What's more, there are still new-comers to Sinclair-Timex computing. The TS1000 surplus is still being doled out into unsuspecting homes, and 2068's are around if you look for them. Other machines are "hand-me-downs" or are picked up at garage sales. Some of these machines will strike that same chord in a brand-new user that it struck in all of us when we first started playing with them. There's something very neat about these little wonders.

Our ace-in-the-hole is that the ZX81 is simple enough to build completely from the ground up, and

could be built from completely stock parts. By looking back at the ZX81 schematic, you realize that this machine will never become a fossil because of blown custom ULA or SCLD chips. (Wags might say it's because it's always BEEN a fossil.)

At any rate, we have a lot going for us. I think that a workable shareware system can be built up. It would involve a lot of work for many members of our community, but would be worth doing for the long-term rewards.

There are a number of ways of implementing this, but here are some comments that just might help to make it work for our little world.

Here's the Plan . . .

The first step is to work up a common data-base of existing programs, cross-referenced and linked to author. In the case of multiple authorship, i.e. "chain" programs, the libraries are allowed a certain discretion in assigning merit for different levels of contribution.

Another massive chore will be to find the authors. This could be tricky in some instances, since many of these guys have given up their Timexes for a Mac at home and IBM's at work. Many have all but forgotten their experience with the ZX81 and Spectrum machines, perhaps because they'd rather not remember how hard they worked and how little they got out of it.

We have to find these people, contact them, and interest them in the shareware notion. Most would be happy to release their "old" material, (it's a cheap way of gaining immortality!) especially if it means a token payment four times a year.

Some of them will regain interest, and will be inspired by the existence of the network into writing new programs. If the new programs are a hit, well then their royalty check will be appropriately bigger over the next few quarters.

Permission to place programs into shareware should be solicited on behalf of the entire library network. Similarly, libraries should adhere to an agreed-on code of ethics regarding pricing, royalty payments, interchange and competition with other libraries.

We have to agree upon the billing formula. How do we keep it fair, yet reasonably simple? How much are users willing to pay, and how little are authors and libraries willing to receive? As a side-note, there could easily be instances of programs costing MORE through shareware, than they did when being retailed in the past.

Some of the other problems we'll run into include topics like media: obviously a tape-dubbing operation is more cost-intensive than if programs are supplied on disk. Note that since the software library gets a percentage based on dubbing and photocopying time, the more efficient its information transferring method is, the better off we'll all be. TS2068 libraries would be at a distinct advantage because of shorter saving/loading times. Perhaps we should adopt a standardized fast-load tape routine for both computer types. How about making TS1000 programs available in TS2068 format? (See Kent Cook's article in SWN 5:1.) Customers buying in this format would get a price-break.

A share-ware library, to be viable, should have all programs rapidly accessible on disk. Which disk standards are supported will depend on which system that each library has available. For instance at this writing I could supply tapes or Compusa disks for ZX81, with Larken disks being the next stage. I could therefore be the librarian for this subset of Timex users.

How do we most effectively use the modem options? Can some if not all of our shareware transactions be conducted via FidoNet or CompuServe?

How do we catalog the entries, and their various versions? How do we agree on a catalog system? Or do we need to?

(As an example of this last point, I have recently decided to catalog two drawers of tapes and a bunch of disordered disks. At this point, my VU-FILE on archived programs has over 400 entries. I estimate that a total of 600 entries will result from just the stuff that I have. [Some of it was even written by me.] Many have multiple versions, of course; the record-holder is Memotext, with some 30 different versions, not counting the help files.)

Assuming that we can cross these hurdles, the overall snowballing effect might just surprise us all. We have discussed this briefly at SWN, and were excited with the possibilities. Imagine . . . all or almost all programs ever written available to anyone, with their creators getting a fair slice of the pie.

I'm sure that I speak for everyone at SyncWare News when I say that we will definitely be involved in some way, perhaps as shareware library co-ordinator. We will try to set up proposed guidelines for shareware library facilities. I'm prepared to be a "node", with my collection of ZX programs, and I'm sure that Jeff, Basil and Tom would be happy to become personally involved at some level also.

You asked us to move you, shake you. We're moving, are you shaking yet? If you felt even the slightest tremor, let us know. This just might be what it

takes to allow our machines to survive, and open the door to new machines that have even more capability.

ShareWare Update

A lot of the real giants in the software-writing field have disappeared--temporarily, we hope. Let's see if we can locate some of them.

Does anyone know where Ray Kingsley is? Hot-Z II and the several Hot-Z 2088 versions will always be in demand. As a long-term pioneer of the share-ware idea, I think he'd be delighted to get a few checks once in a while.

Does anyone know who wrote Memotext? Does anyone know anything about Memotech at all? How to get in touch with John Reidy, who wrote Memocalc?

Anyone with Sinclair connections "across the pond", let yourself be heard. In the U.K. and elsewhere in Europe, the Spectrum family is still very popular. However, even the ZX81 has a small but loyal following. There have been some amazing programs written in Europe, that we have never even heard of, let alone seen in action.

How do we handle programs that appeared in magazines? Suggestions, comments, legal info, would all be much appreciated. Meanwhile, we'll be doing the best we can.

Editor's note:

We'd suggest, though, that you make SNUG your first point of contact (assuming--as we do assume) that they get enough of a response to get "off the ground". This is exactly the sort of thing that SNUG should be doing.

If, however, you feel more comfortable doing business with someone you already know, Fred, Jeff, Tom, and I will be glad to do what we can--even though it might be only to help you to get in touch with SNUG.

GETTING STARTED WITH BETA BASIC

Part 2 - Printer & DOS Routines

SPECTRUM or TS2068 in SPECTRUM mode
Robert D. Hartung

In Part 1 (SWN Vol. 5 No. 5) of this intro to Beta Basic (BB) we looked at some of the editing and data-manipulation features which may be used on programs entered in T/S mode as well as those written in BB. This time let's see how it may be adapted for use with a Spectrum-compatible DOS and also how to install relocatable machine code routines with BB in-residence.

The listings as shown were done in LIST FORMAT 2, using the printer driver routine from Profile re-located into BB to send LLIST output to the printer. Someone may protest, "But the P/F driver doesn't expand tokenwords for an LLIST." True, but BB will do this before sending the LLIST data to a SW driver as well as sending the proper number of TAB n spaces for whatever printer column-width has been set by POKE 57500,n. The USING or USING\$

function allows fixed-digit justification of columnar tabulations on either or both sides of the decimal point for screen displays and printouts. (LPRINT will work OK with the Oliger EPROM driver routine and possibly others, but BB keywords will not be expanded for LLIST.)

Print Driver Installation

To install the P/F driver routine (or any other re-locatable routine) first load the P/F header and code in normal T/S mode. If loading from tape, BREAK before the main listing as the code portion is all we need. To make the P/F driver routine compatible with Spectrum mode, POKE 63688,84 and POKE 63689,31. Make a save of just the P/F driver code block with the direct command SAVE "prcode"CODE 63672,117 (or revise this to define the location and number of bytes used by any other machine code routine).

Now go to Spectrum mode and load Beta Basic. Use your backup copy of the original program for this to make sure RAMTOP has not been moved downward as it will be in copies made after defining windows and user-defined functions that were not erased. As a direct command enter CLEAR 120, which moves RAMTOP down to make room for the driver code block (or CLEAR no. bytes plus 3 if using a driver other than P/F). The extra bytes are provided to prevent the driver code from over-writing BB and BB from over-writing part of the driver under some circumstances. The special BB CLEAR n accepts values of minus or plus 1-767 to move RAMTOP either upward or downward, and does not affect any existing variables, the screen, or the BB stack.

To find the new location of RAMTOP, enter the direct command LET rt=DPEEK(23730). Enter PRINT rt and write down the result. Note that the rt (RAMTOP) address marks the top location in free RAM available to Basic, not the beginning location of memory reserved by CLEAR n, which is rt+1. To move the driver code to this new location enter the direct command LOAD "prcode"CODE rt+1,117 and load the code block previously saved.

The first two lines of the header supplied with your Beta Basic should look something like those in the BB HEADER listing. (Later versions may have different line nos. but will include the same statements.) Following the RANDOMIZE USR 58419 in the header listing (which activates BB) insert the line DPOKE 61081,n in which n is the actual number obtained by adding 1 to that value previously noted for rt. This sets the pointer for BB printer output to the starting address of the driver that has been patched in. This pointer is reset to the ROM driver routine each time BB is activated by RANDOMIZE USR 58419. Revise the remaining lines in the header to the commands used by your printer.

DOS Modifications

DOS users should delete the tape auto-save POKE PEEK... statement between SAVE "Beta Basic" LINE 2 and SAVE "BB"CODE.... Adjust syntax in this line to that used by your system. Use <RUN> to make saves of this new header and the new code. SDOS users, and possibly others, will need to de-activate BB before each save by entering RANDOMIZE USR 59904 to prevent an "Invalid file name" error during the auto-VERIFY (although the save itself will be made OK). Each re-start of BB with RANDOMIZE USR 58419 installs an additional set of XOS, XRG, YOS, YRG graphics scale/range variables, taking up memory. To delete these without affecting other variables as CLEAR would do, follow each RANDOMIZE USR 58419 re-start with the routine in lines 30-40 of the RESTART listing. Note that only the part following the BB re-start may be defined as a procedure, but the entire RESTART sequence could be made a subroutine if more than one call is to be made to it during a program.

More Commands

The REVAL DEVAL listing gives two procedures which appeared in BB newsletter #10. Normal floating-point numbers for GO TO, GO SUB, definitions, string-slicing, etc. are easier to use while entering a listing. Then if <reval> is entered these will all be changed to VAL "n" to conserve memory. They are changed back again by <deval>. Note that these procedures must be installed as lines 1-9 since they are designed to alter all lines from 10 on. After checking them out

with a few test lines, make a save of them for merging with other programs. They may also be used on T/S Basic listings with BB in-residence. (Make a tape copy if your DOS MERGE function conflicts with BB line 0.) Add them to your main listing, then, after testing, key <reval> and <DELETE 1 TO 9> to remove these procedures for maximum free memory. ual than we have room for here.

Another form of the JOIN command, eg: JOIN a\$ TO b\$, allows concatenation of strings or arrays in memory, even when only one byte over their combined length remains free. If you have an array a\$(100,10) that you need to extend to a\$(100,20) then DIM b\$(1,20): JOIN a\$ TO b\$: DIM a\$(1,20): JOIN b\$ TO a\$ will pad each string in a\$ to 20 characters, without data loss. If we wished to insert the entire array a\$ into b\$ so that the first inserted string is the 4th string in the enlarged array b\$ then we could use JOIN a\$ TO b\$(4). One use of this might be in an editing routine to shuffle text around in a word-processing program. The COPY command is very similar in syntax and function but does not remove the copied string from memory as JOIN does.

The INARRAY function searches a specified string array for a target string and returns the number of the first array-element in which it is found, or zero if not found. INSTRING does the same for a single-element string such as a\$ or a\$(1000) except that the character-position is returned. The # "don't care" may be substituted for one or more characters in the search-string. Thus PRINT INSTRING(1,a\$,"SM#TH") would find SMITH, SMYTH, SMATH, etc. in the target string, or in an array if INARRAY were used. Again this could be used with DELETE and JOIN to create a search and replace routine for our hypothetical word-processor.

I hope all this has been more informative than confusing and perhaps has created some interest among those who are not already Beta Basic users. If sufficient reader support is indicated, next time we will take a look at actual routines and procedures using some of the many other BB commands. It would add a great deal to the interest and usefulness of this series if users cared to share procedures they have created. QL procedures also can usually be easily translated to BB and vice versa.

Both Beta Basic 3.0 for 48K Spectrum/emulated TS2068 and BB 4.0 for 128K Spectrum are supplied on one cassette for 15.95 pounds sterling by Betasoft, 24 Wyche Ave., Kings Heath, Birmingham B14 6LQ, England. BB 3.0 by itself is a little less because of its smaller manual. MasterCard is accepted by them for easier currency exchange. I have available a 20K-byte demo giving working examples of about 85 BB commands and functions for \$5 U.S. to cover the cost of the tape, packaging, S&H.

My name and address are:

Robert D. Hartung
2416 N. County Line Road
Huntertown, IN 46748

Please note that the version of Beta Basic used by permission of BetaSoft to drive the demo will NOT work with any other listings, and requires Spectrum mode. It may also be used as a tutorial when used with the normal version of BB 3.0.

BB HEADER

```

1 LET rt=PEEK 23730+256*PEEK
  23731
  SAVE "Beta Basic" LINE 2
  POKE PEEK 23631+256*PEEK 23
    632+2,181
  SAVE "BB" CODE rt+1,65367-rt
  STOP
2 CLEAR rt
  LOAD "BB" CODE
  RANDOMIZE USR 58419
3 DPOKE 61081,46559
  REM Actual value of rt+1,
  not the expression
4 PRINT #0;"Printer: 2040 or
  Centronics? "
  PAUSE 8
  LET p$=INKEY$
5 IF p$="2" THEN DPOKE 61081,
  2548
  ELSE INPUT "Left MARGIN set
  (Up-arrow & your ESC seque
  nce);m$
  LPRINT m$
6 DELETE 1 TO 6

```

BB RESTART

```

10 PRINT "Delay for SDOS VERIF
  Y"
  PAUSE 100
20 RANDOMIZE USR 58419
30 IF PEEK (DPEEK(23627)+32)<>
  184 THEN GO TO 50
40 POKE DPEEK(23627)+32,65
  POKE DPEEK(23627)+33,29
  POKE DPEEK(23627)+34,0
  DELETE a$
  GO TO 30
  REM Removes extra x,yos/rq
  copies after BB re-start

```

REVAL_DEVAL

```

1 DEF PROC reval
  DPOKE 48804,10
  LET a$=CHR$ 14+CHR$ 0+CHR
  $ 0
  get_p
  DO
    LET p=INSTRING(p,MEMORY
    $( ) TO DPEEK(23627)),a
    $)
  EXIT IF p=0
  LET t$="",t=p
  DO
    LET t=t-1
    LET p$=CHR$ PEEK t
    EXIT IF p$<"0" OR p$>"9"
    LET t$=p$+t$
  LOOP
3 IF LEN t$<>0 THEN PRINT
  "ALTERing: ";t$
  LET x=VAL t$
  ALTER (x) TO "VAL "+C
  HR$ 34+t$+CHR$ 34
4 LET p=p+1
  LOOP
  DPOKE 48804,1
  END PROC
5 DEF PROC deval
  DPOKE 48804,10
  FOR s=1 TO 5
    LET a$="VAL "+CHR$ 34+S
    TRING$(s,"#")+CHR$ 34
    get_p
6 DO
  LET p=INSTRING(p,MEMO
  RY$( ) TO DPEEK(23627
  )),a$)
  EXIT IF p=0
  LET t$=""
  FOR n=0 TO s+2
    LET t$=t$+CHR$ PEEK
    (p+n)
  NEXT n
7 LET x=VAL t$(3 TO 2+s
  )
  PRINT "ALTERing: ";t$

```

ALTER (t\$) TO (x)

```

  LET p=p+1
  LOOP
8 NEXT s
  DPOKE 48804,1
  END PROC
9 DEF PROC get_p
  LET p=DPEEK(23637)
  END PROC

```

MAIL FILE

```

1 REM Insert REVAL_DEVAL pro-
  cedures here (lines 1-9)
10 PRINT "Delay for SDOS VERIF
  Y"
  PAUSE 100
20 RANDOMIZE USR 58419
30 IF PEEK (DPEEK(23627)+32)<>
  184 THEN GO TO 50
40 POKE DPEEK(23627)+32,65
  POKE DPEEK(23627)+33,29
  POKE DPEEK(23627)+34,0
  DELETE a$
  GO TO 30
  REM Removes extra x,yos/rq
  copies after BB re-start
50 LET i=SGN PI,o=NOT PI,y$=""
  ds=""
60 CLS
  GO TO 470
  REM To main menu
70 INPUT "EDIT y/n? ";e$;" LPR
  INT y/n? ";y$;"ID/phone nos
  . y/n? ";d$
80 INPUT "CHR$ in search-word
  (#=wild): ";c$
90 FOR n=1 TO last
100 LET p$=f$(n)
110 IF INSTRING(1,p$,c$) THEN
  fprint y$,f$(n),i,o
  IF e$="y" THEN fedit f$
  (n),last,i,o
120 NEXT n
130 PRINT #0;"Search is complet
  e"
  PAUSE 300
  GO TO 470
140 REM INPUT
150 INPUT "ALL NEW & ERASE y/n?
  ";y$
  IF y$<>"y" THEN GO TO 180
160 INPUT "Maximum no. of new e
  ntries<150? ";E
  IF E>150 THEN LET E=150
170 DIM f$(E,115)
180 FOR n=1 TO E
190 IF f$(n,2)<>" " AND f$(n,
  114)<>"#" THEN NEXT n
200 PRINT INVERSE I;n
  INPUT " STOP iMr/Mrs 2Mr
  3Mrs 4Ms 5Miss 6att: OR
  9 then title or first nam
  e ";p$
  PRINT p$;
  LET f$(n, TO 20)=p$,f$(n,
  115)=CHR$ LEN p$
  IF p$=" STOP " THEN LET f
  $(n)=""
  GO TO 290
210 INPUT (f$(n, TO 20))/"Las
  t name"/f$(n,21 TO 40)/"A
  ddress"/f$(n,41 TO 65)/"P
  .0. or 1AUB, 2BUT, 3HNT"/
  f$(n,66 TO 80)/"State (2
  chr$+2 spaces) & ZIP"/f$(
  n,81 TO 94)/"IDCODE OR ph
  one no. TO re-enter"/f$(n
  ,95 TO 114)
220 IF f$(n,66)="1" THEN LET
  f$(n,66 TO 80)="Auburn",f
  $(n,81 TO 94)="IN 46706"
230 IF f$(n,66)="2" THEN LET
  f$(n,66 TO 80)="Butler",f
  $(n,81 TO 94)="IN 46721"
240 IF f$(n,66)="3" THEN LET
  f$(n,66 TO 80)="Huntertow
  n",f$(n,81 TO 94)="IN 46
  748"
250 REM P.O. auto-entries he
  re

```

```

260 PRINT " :f$(n,21 TO 94) '
    f$(n,95 TO 114)
270 IF f$(n,95)=" TO " THEN G
    O TO 200
280 NEXT n
290 LET last=n-1
    SORT f$(1 TO last)(21 TO 40
    )
    GO TO 470
300 REM EDIT
310 CLS
320 FOR n=1 TO last
330   fprint y$,f$(n),I,o
340   fedit f$(n),last,I,o
350 NEXT n
360 SORT f$(1 TO last)(21 TO 40
    )
370 LET x=last
380 FOR n=x-z TO x
390   IF f$(n,21 TO 30)="zzzzzz
      zzzz" THEN LET last=last-
      I,f$(n)="
400 NEXT n
410 GO TO 470
420 REM LPRINT
430 INPUT " PRINT ID/phone nos.
    y/n? ";d$
    LET y$="y"
440 FOR n=1 TO last
450   fprint y$,f$(n),I,o
460 NEXT n
470 CLS
    PRINT "1 - Start all new fi
    le""2 - Add entries""3 -
    List/edit""4 - Search/prin
    t/edit""5 - LPRINT ""6 -
    SAVE ""7 - LOAD "" FLASH I
    ""B board OFF for SDOS"
480 PAUSE o
490 ON ERROR 470
    LET p=VAL INKEY$
    CLS
    ON ERROR o
500 GO TO ON p;160,180,300,70,4
    20,520,540
510 GO TO 470
520 CAT /
    INPUT FLASH I;"B board ON";
    FLASH o;" & enter SAVE name
    : ";c$
    RANDOMIZE USR 59904
    SAVE /c$ LINE 10
530 GO TO 10
540 CAT /
    INPUT "LOAD name? ";c$
    RANDOMIZE USR 59904
    LOAD /c$
550 GO TO 10
560 DEF PROC fprint y$, REF f$,
    I,o
570   IF y$="y" THEN OPEN #2,"P

```

```

580 PRINT ("Attention of:" AN
    D f$(n,I)="6")
    REM Or other repeat titl
    e
590 PRINT ("Mr. & Mrs." AND f
    $(n,I)="1")+("Mr." AND f$
    (n,I)="2")+("Mrs." AND f$
    (n,I)="3")+("Ms." AND f$(
    n,I)="4")+("Miss" AND f$(
    n,I)="5")+(" " AND f$(n,I
    )<"6");
600 PRINT f$(n,2 TO CODE f$(n
    ,115));" "
610 PRINT f$(n,21 TO 40)
620 IF f$(n,41)<>" " THEN PRI
    NT f$(n,41 TO 65)
630 IF f$(n,66)<>" " THEN PRI
    NT f$(n,66 TO 80)
640 IF f$(n,81)<>" " THEN PRI
    NT f$(n,81 TO 94)
650 IF d$="y" AND f$(n,95)<>"
    " THEN PRINT f$(n,95 TO
    114)
660 PRINT
670 CLOSE #2
680 IF y$="y" THEN PRINT #o;A
    T o,o;"Any key for next e
    nvelope"
    PAUSE o
    INPUT ""
690 END PROC
700 DEF PROC fedit REF f$,last,
    I,o
710   LET z=o
720   LET p=f$(n)
730   INPUT "NL next, # DELETE,
    TO keep, STOP or space:
    ;(f$(n, TO 20));TAB 9;d$
    IF d$=" " THEN GO TO 820
740   IF LEN d$=2 THEN LET f$(
    n, TO 20)=d$(2 TO ),f$(n,
    115)=CHR$ LEN d$(2 TO )
750   IF d$=" " STOP " THEN GO TO
    360
760   IF d$="#" THEN LET f$(n,2
    1 TO 40)="zzzzzzzzzz"
    LET z=z+1
    GO TO 820
770   EDIT f$(n,21 TO 40)
780   EDIT f$(n,41 TO 65)
790   EDIT f$(n,66 TO 80)
800   EDIT f$(n,81 TO 94)
810   EDIT f$(n,95 TO 114)
820 END PROC

```

BASIL'S COMPENDIUM

"Painless" MC Development

Basil Wentworth

This chapter will give you a look into "real life", and will show you how to make a "rough draft" of your programs in BASIC.

I always find it easier to debug my programs in BASIC before actually translating them into machine code, as the results of an error are much less drastic in BASIC. However, I try to make my BASIC program look as much like the final machine code as I can, in order to minimize errors in translation. This results in a far-from-efficient BASIC program, of course. You'd have much the same situation if you tried to write a sentence in English so that you could translate it word for word into correct Spanish. Your final product could look very much like: "Themselves to me they lost my pencils".

But even before the rough draft, you'll have to have an idea of where you are going. Here you'll find the table called "The System Variables" very useful. This table begins on page 134 in the ZX81 manual, and on page 262 for the TS 2068.

So What To Do?

Let's start out with that famous line renumbering program. For this application, you should remember the following facts:

1. The first byte of the program is at 16509 for the ZX81. For the TS2068, the location of the first byte is given by

PEEK 23635 + 256 * PEEK 23636

which is 26710, unless you have POKEd 23635 or 23636.

2. The end of each line is marked by 118 in the ZX81, by 13 in the 2068.

3. Each line invariably starts with a sequence of the following four bytes:

a. Two bytes giving the line number in 256-imal, MOST significant byte first.

b. Two bytes giving the line length in 256-imal, LEAST significant byte first. This length is the number of all bytes FOLLOWING the line-length bytes, INCLUDING the 118 (or 13) at the end of the line. It does not, of course, include the bytes marking the line number nor does it include the line-length bytes.

To illustrate, write the very short program

```
10 LET BC=10
```

and PEEK the first 17 bytes. You will see:

0	0
10	10
13	13
0	0
.	.
(. .12 more bytes. .)	
.	.
13	118

for the 2068 and the ZX81, respectively. The two digits 13 and 0, in 256-imal (least significant byte first), give the number of bytes remaining in the line, INCLUDING the line-end indicator 13 or 118. The two bytes 0 and 10 give the line number, MOST significant byte first.

Now to put our knowledge to work. Remember that this program will foul up all GO TOs and GO SUBs, unless you carefully design the program (as we will) to avoid that problem.

First, you must decide which register to use for each variable that you have to work with. The most significant factor affecting your choice will be that you can do things with HL (a 2-byte register) and A (one byte) that you can't with the others. They are the true workhorses of the computer. The next consideration is that instruction 235 provides a one-byte provision for exchanging DE and HL. This makes DE the best choice for an alternate workhorse. BC gets whatever is left.

Now, what we're going to do in the line-renumbering program is to look at each byte in succession. When we reach an end-of-line, we pay special attention to the next two bytes (the line number), POKEing in the number that we have planned for that location. We have to include some provision for stopping the program when we're finished. And we mustn't forget the RETURN to BASIC when we finally get around to writing the machine code routine. We'll leave the first program line number unchanged, at 1. This might very well be the place where we store the machine code in the final version.

Apparently, we're going to have to search through the memory, byte by byte. HL seems a logical choice for this function. Then we're going to have to keep a record of what the next line number must be. A logical use for DE. BC, then, can remember the interval between line numbers--a value that will not change. And A will be useful to set the target,

as it were, for each phase of our search. Register A is a particularly happy choice for this purpose, because of the existence of the CP instruction.

For the ZX81

Let's take the ZX81 case first. Start with the following program:

```
1 REM
10 LET HL=16509
20 LET BC=10
   note: interval between line numbers
30 LET DE=10
   note: number of first line
40 LET A=117+1
   note: remember that 118 is a "no-no"
50 LET HL=HL+1
60 LET CP=A-PEEK HL
70 IF CP<>0 THEN GOTO 50
80 PRINT 256*PEEK (HL+1)+PEEK
(HL+2)
```

Now RUN this much of the program. The number PRINTed should be 10. Insert (temporarily)

```
2 REM
```

and RUN again. The result should be 2. DELETE line 2. Now add

```
90 LET A=35
100 LET HL=HL+1
110 LET CP=A-PEEK HL
120 IF CP<=0 THEN STOP
   note: will be RETURN in M/C routine
   note: this will stop the program as soon as
the line number under consideration exceeds 8960
(=35*256)
210 GOTO 40
```

Now RUNNING the program will give a list of all line numbers from 2 to the first one over 8960. Experiment by inserting other REM lines, and RUNNING again. Be careful, however, that you DELETE all the extra lines, and restore the program to its present form before going on to the next step. This is so that the GOTOs will still work after the renumbering process is carried out.

Now continue:

```
130 LET D=INT (DE/256)
140 LET E=DE-256*D
150 POKE HL,D
160 LET HL=HL+1
170 POKE HL,E
171 STOP
```

Now another test. Change line number 10 to 2: then RUN. If you have followed instructions correctly, the line number will be changed back to 10 again. Now we're coming down the homestretch. DELETE line 171, and add lines 9000 to 9030 to provide a subroutine that simulates the function that exchanges DE and HL:

```
9000 LET TEMP=HL
9010 LET HL=DE
9020 LET DE=TEMP
9030 RETURN
and
180 GOSUB 9000
190 LET HL=HL+BC
200 GOSUB 9000
```

And line 210, of course, still reads GOTO 40.

For a final test, add a few REM lines between 211 and 8960. You may change line numbers lower than 210, if you wish, but make sure that the GOTOs and GOSUBs will be valid BOTH BEFORE AND AFTER the renumbering. Any numbers above 8960 ($=256*35$) will be unchanged. And, of course, you can change this limit to any multiple of 256 that you wish. You can also set it to other limits, but the procedure takes a couple more bytes of machine language program.

For convenience, I'd suggest that you change line 80 to a REM (but don't DELETE it, lest you mess up the GOTO addresses). If you'd prefer to keep it, then you'll have to CONTINUE after the screen fills up with line numbers.

For the TS2068

The 20608 procedure is a little bit more scary, because, as you have seen, you can run into 13's that aren't end-of-line indicators. (Theoretically, you could have this problem with the ZX81, but only if you run into a line long enough so that the line-length indicator contains a 118). With that one warning, the 2068 procedure follows very closely that of the ZX81, like this:

```
1 REM
10 LET hl=PEEK 23635+256*PEEK
23636
20 LET bc=10
30 LET de=10
40 LET a=13
50 LET hl=hl+1
60 LET cp=a-PEEK hl
70 IF cp<>0 THEN GO TO 50
80 PRINT 256*PEEK (hl+1)+PEEK
(hl+2)
210 GO TO 40
```

When you RUN this program, you get all of the line numbers, but there are also spurious responses after 20, 30, and 40. The first result from the fact that lines 20 and 30 each have indicated line length of 13 bytes; line 40 gives trouble because the 13 is explicitly expressed. The search through all the hl's reads these 13's as if they were ends of lines.

This problem can be corrected by changing lines 20 through 40 as follows:

```
20 LET bc=9+1
30 LET de=9+1
40 LET a=12+1
```

Now fill in the rest of lines 90 through 200 as above.

The translation to machine code should be very straightforward. Just remember that the POKE and PEEK functions are represented by parentheses, like

LD (HL),N LD hl,(NN)

and the like.

An Unorthodox Application

And now, what do you do with such a routine? Well, of course, you could use it to clean up your programs (taking due care with GO TO and GO SUB addresses). But here is another application that might not have occurred to you:

Suppose you are preparing an index--say, of composers. You may want a simple LISTing, in which case you just put the names in REM lines, or you may want it to provide printout, in which case you use the command PRINT. You start out confidently like this:

```
10 REM BACH
20 REM BEETHOVEN
30 REM BRAHMS
```

BERLIOZ can go in as line 25: BORODIN as 27. BERLIN would take line 22. Maybe BASIL in line 15. Pretty soon, your list looks like this:

```
2 REM BACH, CPE
5 REM BACH, JC
10 REM BACH, JS
15 REM BASIL
20 REM BEETHOVEN
22 REM BERLIN
25 REM BERLIOZ
26 REM BIZET
27 REM BORODIN
30 REM BRAHMS
```

But HEY! Where are you going to put Bordogni? Or Seth Bingham? And what to do with the rest of that huge Bach tribe?

Easy. Just renumber, and the above set will look like this:

```
10 REM BACH, CPE
20 REM BACH, JC
30 REM BACH, JS
40 REM BASIL
50 REM BEETHOVEN
60 REM BERLIN
70 REM BERLIOZ
80 REM BIZET
90 REM BORODIN
100 REM BRAHMS
```

Plenty of room for insertions, even when a Billings or a Birmingham turns up.

That's all for now. There will be more.

.....

UPDATE!



A Quarterly Magazine for the users

Supporting the Sinclair QL, Z88, and TS-2068
Subscription \$15.00 Year. UPDATE Magazine,
1317 Stratford Ave., Panama City, FL 32404

Last issue, *SyncWare News* printed a simple flow chart generating program. Our Editor issued the following challenges to our readers: 1). Reduce the memory required by the example program to the barest minimum; 2). Create a machine code equivalent of the example program, and/or 3). Write an

equivalent program for the ZX81/TS1000 in either MC or BASIC.

The following two programs were submitted in response to the challenge. We're still looking for an equivalent program for the ZX81/TS1000.

THE 'OLD GENTS' FLOWCHARTER

"The Old Gent"
TS2068

The author of this challenge response prefers to be known by his handle, rather than by his name. So who am I to argue?

A few things worth knowing . . .

A couple of comments on the program would be in order. First, the program will not work if you have any material tucked away down below the byte defined by

```
PEEK 23635+256*PEEK 23636
```

(PROG in the list of Systems Variables in the manual). I discovered this the hard way, as my printer driver routine resides down in that area. Or rather, it will interpret the material you have cached there as program, and will try to flowchart it. The results can be out-of-the-ordinary, to put it mildly.

FLOWCHART 2068 CHALLENGE
BY THE OLD GENT
(Ref: SYNCWARE NEWS V5/N6 pg.16)

PROGRAM FEATURES:-

- *Automatic print to TS2040
- *Graphic SYMBOLS vs plot/draw
- *Alpha constants for all numbers
- String constants for SYMBOLS
- ie:- d>z,dd>jj,a\$>g\$
- a>c=program variables

INSTRUCTIONS:-

- *Turn the PRINTER on !!
- *PRESS j to LOAD FLOWCHART
- *To reprint type GO TO 1:ENTER
- NOTE-reLOAD if you RUN or CLEAR
- *To copy program LLIST:ENTER
- *To print chart on the screen
- type OPEN #3,"s":GO TO 1:ENTER

PRESS z TO COPY THIS SCREEN

"OLD GENT's" Loader Program:
Initial Display

This deficiency, if it bothers you, could be remedied by defining the starting address (line 9902) as shown above, rather than as an explicit 26710 (=d in the Old Gent's list of pre-set variables).

Second, it would have been useful to include a RESTORE in the LOADER program. Without the RESTORE, the program stalls with an Out of DATA message if you GO TO 1 a second time, as advised in line 1025.

Third, a small note--it would have helped absent-minded people like me if the LOADER program had included a "start tape" message (with BEEPs) after "j" is pressed (line 1020).

This program uses the standard technique of pre-defining variables (as shown in the Table) to eliminate the memory consumption that would otherwise result from using program space to define them. The price paid for this technique is, as the program tells us, the fact that you can not CLEAR or RUN the program without losing all of these values. Use GO TO 1 instead of RUN.

The program uses LPRINT for display commands. As indicated, the command OPEN #3,"s" disables the printer, and translates the commands to screen display instead. If you want to re-enable the printer, key in "CLOSE #3", then ENTER.

Pre-defined variables

00	00000000	00	10
01	00000000	01	00000000
02	00000000	02	00000000
03	00000000	03	00000000
04	00000000	04	00000000
05	00000000	05	00000000
06	00000000	06	00000000
07	00000000	07	00000000
08	00000000	08	00000000
09	00000000	09	00000000
10	00000000	10	00000000
11	00000000	11	00000000
12	00000000	12	00000000
13	00000000	13	00000000
14	00000000	14	00000000
15	00000000	15	00000000
16	00000000	16	00000000
17	00000000	17	00000000
18	00000000	18	00000000
19	00000000	19	00000000
20	00000000	20	00000000
21	00000000	21	00000000
22	00000000	22	00000000
23	00000000	23	00000000
24	00000000	24	00000000
25	00000000	25	00000000
26	00000000	26	00000000
27	00000000	27	00000000
28	00000000	28	00000000
29	00000000	29	00000000
30	00000000	30	00000000
31	00000000	31	00000000
32	00000000	32	00000000
33	00000000	33	00000000
34	00000000	34	00000000
35	00000000	35	00000000
36	00000000	36	00000000
37	00000000	37	00000000
38	00000000	38	00000000
39	00000000	39	00000000
40	00000000	40	00000000
41	00000000	41	00000000
42	00000000	42	00000000
43	00000000	43	00000000
44	00000000	44	00000000
45	00000000	45	00000000
46	00000000	46	00000000
47	00000000	47	00000000
48	00000000	48	00000000
49	00000000	49	00000000
50	00000000	50	00000000
51	00000000	51	00000000
52	00000000	52	00000000
53	00000000	53	00000000
54	00000000	54	00000000
55	00000000	55	00000000
56	00000000	56	00000000
57	00000000	57	00000000
58	00000000	58	00000000
59	00000000	59	00000000
60	00000000	60	00000000
61	00000000	61	00000000
62	00000000	62	00000000
63	00000000	63	00000000
64	00000000	64	00000000
65	00000000	65	00000000
66	00000000	66	00000000
67	00000000	67	00000000
68	00000000	68	00000000
69	00000000	69	00000000
70	00000000	70	00000000
71	00000000	71	00000000
72	00000000	72	00000000
73	00000000	73	00000000
74	00000000	74	00000000
75	00000000	75	00000000
76	00000000	76	00000000
77	00000000	77	00000000
78	00000000	78	00000000
79	00000000	79	00000000
80	00000000	80	00000000
81	00000000	81	00000000
82	00000000	82	00000000
83	00000000	83	00000000
84	00000000	84	00000000
85	00000000	85	00000000
86	00000000	86	00000000
87	00000000	87	00000000
88	00000000	88	00000000
89	00000000	89	00000000
90	00000000	90	00000000
91	00000000	91	00000000
92	00000000	92	00000000
93	00000000	93	00000000
94	00000000	94	00000000
95	00000000	95	00000000
96	00000000	96	00000000
97	00000000	97	00000000
98	00000000	98	00000000
99	00000000	99	00000000
100	00000000	100	00000000

```
900 CLS : FOR a=0 TO 18 STEP 2
902 BEEP .1,20: BEEP .1,16: BEE
P .1,12
904 PRINT AT a,a;"STOP THE TAPE
"
906 PAUSE 10: NEXT a
910 FOR b=65368 TO 65471
911 READ c: POKE b,c: NEXT b
912 DATA 31,16,32,32,64,64,128,
255
```



```

913 DATA 255.1,2,2,4,4,8,248
914 DATA 255.128,128,128,128,12
8,128,255
915 DATA 255.1,1,1,1,1,1,255
916 DATA 63,64,128,128,128,128
,64,63
917 DATA 252.2,1,1,1,1,2,252
918 DATA 0,0,4,2,255,2,4,0
919 DATA 1,6,24,96,128,96,24,6
920 DATA 0,192,48,12,3,12,48,19
2
921 DATA 7,8,8,8,8,8,8,7
922 DATA 192,32,32,32,63,32,32,
192
923 DATA 1,1,1,1,1,5,3,1
924 DATA 0,0,0,0,0,64,128,0
1000 CLS : PRINT AT 0,4;"FLOWCHA
RT 2068 CHALLENGE";AT 1,8;"BY TH
E OLD GENT""(Ref: SYNCWARE NEWS
V5/N6 pg.16)""
1005 PRINT "PROGRAM FEATURES:-
      *Automatic print to
TS2040      *Graphic SYMBOLS vs
plot/draw"
1010 PRINT "*Alpha constants for
all numbers String constants fo
r SYMBOLS      ie:- d>z,dd>jj,a$>
g$      a>c=program v
ariables"
1015 PRINT "INSTRUCTIONS:-"
1020 PRINT "*Turn the PRINTER on
!!      *PRESS j to LOAD FLO
WCHART"
1025 PRINT "*To reprint type GO
TO 1:ENTER NOTE-reLOAD if you
RUN or CLEAR*To copy program LL
IST:ENTER"
1030 PRINT "*To print chart on t
he screen type OPEN #3,""s""
:GO TO 1:ENTER"
1060 PRINT #0;AT 1,0;"PRESS z TO
COPY THIS SCREEN": PAUSE 0
1070 IF INKEY$="z" THEN COPY
1075 IF INKEY$<>"j" THEN BEEP
1,10: GO TO 1060
1080 CLS : LPRINT "FLOWCHART": L
PRINT : PRINT : FLASH 1;"LOADING
": LOAD "FLOWCHART"

```

"Old Gent's" flowchart--loader

```

1 CLS
10 IF 5>1 THEN GO TO 9000
20 FOR f=1 TO 3: PRINT 5: NEXT
f
30 GO SUB 400
40 LOAD ""
50 PRINT 1
60 STOP
70 LOAD "a"
80 SAVE "a" LINE 10: BEEP .3,1
2: BEEP 1,12: VERIFY "a": BEEP .
3,16: BEEP 1,16
90 NEW
100 LIST 10
350 STOP
400 PRINT 123: RETURN
410 STOP
9902 LET a=d
9903 LET c=PEEK (a+n)+q*PEEK a:
LET a=a+p: IF c>z THEN STOP
9904 LPRINT c;TAB p;CHR$ PEEK a;
9905 LET b=PEEK a: IF b=gg OR b=

```

```

jj OR b=ff THEN LPRINT ,c$: LPR
INT : LET a=a+o: GO TO w
9907 IF b=ii OR b=i THEN LPRINT
,d$;: LET a=a+k: GO TO f
9909 IF b=hh THEN LPRINT ,b$: G
O TO x
9910 IF b=h THEN GO TO i
9912 LPRINT ,a$: IF b=g OR b=h T
HEN GO TO u
9914 LPRINT ,f$
9916 LET b=PEEK a: IF b=r THEN
GO SUB e
9917 IF b=s THEN LET a=a+p
9918 IF b=dd THEN LET a=a+n: GO
TO w
9919 IF b=m THEN LET a=a+n: LPR
INT g$;CHR$ PEEK a;: GO TO t
9920 LET a=a+n: GO TO u
9926 IF b=i THEN LPRINT ,,f$: G
O TO w
9927 GO SUB y: IF b=s THEN LET
a=a+p
9929 IF b<>ee THEN GO TO j
9930 GO SUB y: LPRINT " ";CHR$ b
;: IF b=g OR b=h THEN GO TO v
9932 GO TO x
9933 LPRINT ,e$;
9934 GO SUB y: IF b<>s THEN GO
TO v
9936 LET a=a+k: LPRINT PEEK a+q*
PEEK (a+n);: GO TO x
9951 GO SUB y: IF b=r THEN RETU
RN
9952 GO TO e
9960 LET a=a+n: LET b=PEEK a: RE
TURN

```

"Old Gent's" Flowchart Program

FLOWCHART

```

1 CLS
10 IF 5>1 THEN GO TO 9000
20 FOR f=1 TO 3: PRINT 5: NEXT
f
30 GO SUB 400
40 LOAD ""
50 PRINT 1
60 STOP
70 LOAD "a"
80 SAVE "a"
: BEEP
: BEEP
: VERIFY
: BEEP
: BEEP
90 NEW
100 LIST
350 STOP
400 PRINT
: RETURN
410 STOP

```

"Old Gent's" FLOWCHART

DUNNINGTON FLOWCHARTER

Earl Dunnington
TS2068

(Editor's NOTE: This article would be worthwhile reading if it didn't contain anything but the material on programming above line 9999.)

My version of the BASIC program uses 1004 bytes after deleting all lines less than 9000 and ENTERING CLEAR in accordance with the CHALLENGE instructions.

Our Editor did not like the treatment of the NEW command and as I agree, I changed this to use the start stop symbol and dropped the program continuation line and arrow. I also changed the GOSUB command treatment to add the program continuation line and arrow. The treatment of the RETURN command was changed to use the circle with an arrow to the side.

To save four bytes for each line number used, I combined lines 9903 & 9904; 9906 & 9907; 9912 & 9913; 9925 & 9926; 9927 & 9928; 9930 & 9931; 9936, 9937 & 9938; 9940 & 9941; 9946 & 9947.

LET e=22 in line 9946 was deleted as the variable e was not used in the program. I deleted the INPUT ;: in line 9944 as useless. Because this is a utility program, to be MERGED, I feel that the use of color is a waste of memory. Therefore, I deleted BRIGHT SGN PI in line 9942. To save the eyesight of the bifocal group, of which I am one, Caps were used whenever possible. I normally program with the CAPS LOCK on.

A single letter variable occupies six and a double letter variable seven bytes in the VARS area. The rules of the CHALLENGE do not state that the program must be operable after CLEAR has been ENTERED which clears the VARS area, deleting all of the variables, string variables and arrays. By assigning all of the numerical constants in the program to a single or double letter variable entered either as a direct command or in lines to be deleted after RUNNING the line, you can save a lot of memory in the PROG area of the computer. It then requires only one or two bytes in the program area for any numerical constant including GO TOs and GOSUBs. When the program is SAVED the assigned variables are also saved, even without any lines. After LOADING the variables, the program lines can be MERGED to form the complete program to be SAVED. Assigning the PRINT strings, lines 9940, 9942, 9948 to string variables, saves some more bytes in the program area.

I assigned the variables as follows:

TABLE OF VARIABLES

E =-PI	: V =+16	: AL=227	: AZ=253	: BN=10020
H = 0	: W =+19	: AM=230	: BA=254	: BO=10021
I =+1	: Z =+22	: AN=231	: BB=256	: BP=10024
J =+2	: AA=-24	: AO=232	:	: BQ=10025
K =+3	: AB=+26	: AP=234	: BD=10000	: BR=10026
M =+4	: AC=-28	: AQ=235	: BE=10001	: BS=10029
N =-4	: AD=-32	: AR=236	: BF=10003	: BT=10030
O =+5	: AE=+34	: AS=237	: BG=10008	: BU=10031
P =-7	: AF=+58	: AT=241	: BH=10010	: BV=10032
Q =-8	: AG=+64	: AU=242	: BI=10011	: BW=10035
R =+8	: AH=130	: AV=243	: BJ=10016	: BX=10036

```
S =+12 : AI=175 : AW=247 : BK=10017 : BY=10038
T =+13 : AJ=203 : AX=249 : BL=10018 : BZ=23635
U =+14 : AK=226 : AY=250 : BM=10019 : CA=23636
A$="" "c" TO CONTINUE ; "p" TO PRINT
B$=" END OF BASIC PROGRAM"
C$=" : "
```

When developing utility programs in BASIC designed to be MERGED with other programs, I like to use line numbers 10000 and above. This way there will be no conflict of numbers between the two programs. Line numbers of 10000 to 10999 are represented by the computer as :ddd where the colon is ten thousand and each d is a digit. Line numbers of 10000 or above can not be ENTERED directly but must be POKED into the first two addresses of a stored program line, the High Byte then the Low Byte.

The High Byte for 10000=INT (10000/256)=39

The Low Byte for 10000=10000-256*HB=16

The line can not be EDITed after it has been changed. GO TO or GO SUB 10000 etc can be ENTERed or used in a program line and will operate normally as will RETURN.

In the program area, a numerical value of 10000 would use 11 bytes. The substitution of VAL "10000" uses eight bytes. Representing 10000 by a double letter variable uses only two bytes in the program area.

To program my version of "Flowchart" using a tape recorder (see NOTE below):

1. Type in, RUN, and SAVE the "Assignment of Variables" program. Do not delete the lines before SAVING as you may have an error that will show up later.
2. Use NEW to clear the computer.
3. Type in and SAVE the combined "ReNUMBER and Preliminary Flowchart" program. DO NOT RUN BEFORE SAVING.
4. Enter RUN, then delete lines 420 to 510 and enter CLEAR.
5. MERGE the "Assignment of Variables" program and delete lines 1, 2, and 3.
6. SAVE the final version of "Flowchart".

To test the program:

1. Turn the computer off and then on.
2. Type in the Demo lines 1 through 410.
3. MERGE the final version of "Flowchart", and CLS.
4. Use GO TO 10000 to operate the program.
5. If you inadvertently use RUN or CLEAR, just MERGE the final version of "Flowchart" again.

NOTE: The Larken Disk Operating System (LKDOS) V3 does not load the variables when MERGE is used. In programming step 5 after MERGEing the Assignment of Variables, you would have to add line 4 STOP and then RUN the program and delete lines 1 to 4. In testing the program step 2 you would SAVE the Demo

lines program, after typing them in. In step 3 you would LOAD the final version of Flowchart and MERGE the Demo lines.

```
1 REM Assignment of Variables
  by EARL DUNNINGTON
4356 King Theodore Dr.,
Boynton Bch., FL 33436
phone 407-732-6219
```

```
2 LET E=-PI: LET H=0: LET I=1
: LET J=2: LET K=3: LET M=4: LET
N=-4: LET O=5: LET P=-7: LET Q=
-8: LET R=8: LET S=12: LET T=13:
LET U=14: LET V=16: LET W=19: L
ET Z=22: LET AA=-24: LET AB=26:
LET AC=-28: LET AD=-32: LET AE=3
4: LET AF=58: LET AG=64: LET AH=
130: LET AI=175: LET AJ=203: LET
AK=226: LET AL=227: LET AM=230:
LET AN=231: LET AO=232: LET AP=
234: LET AQ=235: LET AR=236: LET
AS=237: LET AT=241: LET AU=242:
LET AV=243: LET AW=247: LET AX=
249: LET AY=250: LET AZ=253: LET
BA=254: LET BB=256
```

```
3 LET BD=10000: LET BE=10001:
LET BF=10003: LET BG=10008: LET
BH=10010: LET BI=10011: LET BJ=
10016: LET BK=10017: LET BL=1001
8: LET BM=10019: LET BN=10020: L
ET BO=10021: LET BP=10024: LET B
Q=10025: LET BR=10026: LET BS=10
029: LET BT=10030: LET BU=10031:
LET BV=10032: LET BW=10035: LET
BX=10036: LET BY=10038: LET BZ=
23635: LET CA=23636: LET A$=""c
"" TO CONTINUE ;""p"" TO PRINT":
LET B$="" END OF BASIC PROGRAM":
LET C$="" :
```

ASSIGNMENT OF VARIABLES PROGRAM

```
420 REM Renumber and Preliminary
Flowchart program
430 LET START=(PEEK 23627+256*P
EEK 23628)-1005: REM Address of
byte before line 9901
440 LET HIGHBYTE=39: LET LOWBYTE
E=16: REM 10000=39*256+16
450 FOR N=START TO START+1004-1
: REM 1004 is number of bytes in
flowchart
460 IF PEEK N<>13 THEN GO TO 5
00: REM if not end of line chara
cter go to next address
470 POKE N+1,HIGHBYTE: POKE N+2
,LOWBYTE: REM Poke new line numb
er
480 LET LOWBYTE=LOWBYTE+1: REM
advance new line number by one
490 LET N=N+5: REM Skip over ad
dresses of line number and lengt
h of line
500 NEXT N
510 STOP
9901 LET A=PEEK BZ+BB*PEEK CA: L
ET C=H: LET Y=A: LET X=AH
9903 LET L=PEEK (A+1)+BB*PEEK A:
LET A=A+M: IF L>=BD THEN GO TO
BX
9905 PRINT L: TAB 0: CHR$ PEEK A:
LET C=C+J: PRINT
9906 PLOT X,Y: LET B=PEEK A: IF
```

```
B=AK OR B=AM OR B=AP OR B=AU THE
N GO TO BL
9908 IF B>AL AND B<AN OR B=AO OR
B=AQ OR B=AT OR B=AW OR B=AX OR
B=AZ THEN GO TO BM
9909 IF B=AV OR B=AY THEN GO TO
BN
9910 IF B=AR OR B=AS OR B=BA THE
N GO TO BQ
9911 DRAW V,H: DRAW N,Q: DRAW AC
,H: DRAW M,R: DRAW S,H
9912 LET Y=Y-R: IF B=AK OR B=AR
OR B=AM OR B=BA THEN GO TO BH
9914 PLOT X,Y: DRAW H,P: DRAW M,
M: DRAW N,N: DRAW N,M
9915 LET Y=Y-R
9916 LET B=PEEK A: IF B=AE THEN
GO SUB BY
9917 IF B=U THEN LET A=A+M
9918 IF B=T THEN GO TO BJ
9919 IF B=AF THEN GO TO BS
9920 LET A=A+I: GO TO BI
9921 IF C=Z THEN GO TO BU
9922 LET A=A+I: GO TO BE
9923 DRAW S,H: DRAW M,Q,E: DRAW
AA,H: DRAW N,R,E: DRAW S,H: GO T
O BG
9924 DRAW V,H: DRAW H,Q: DRAW AD
,H: DRAW H,R: DRAW V,H: GO TO BG
9925 DRAW R,N: DRAW R,H: DRAW N,
M: DRAW M,N: DRAW N,N: DRAW M,M:
DRAW Q,H: DRAW Q,N: DRAW Q,M: D
RAW R,M: IF B=AV THEN GO TO BP
9927 LET A=A+I: LET F=PEEK A: IF
F=U THEN LET A=A+M
9929 IF F<>AJ THEN GO TO BO
9930 LET A=A+I: LET F=PEEK A: PR
INT AT C-J,W:CHR$ F: PRINT : IF
F=AR OR F=AS THEN GO TO BR
9932 GO TO BG
9933 CIRCLE X,Y-M,M: PLOT X+M,Y-
M: DRAW AG,H: DRAW N,M: DRAW M,N
: DRAW N,N: PLOT X,Y: IF B=BA TH
EN GO TO BG
9934 LET A=A+I: LET F=PEEK A
9935 IF F<>U THEN GO TO BR
9936 LET A=A+K: LET G=PEEK A+BB*
PEEK (A+1): PRINT AT C-J,AB:G: P
RINT : GO TO BG
9939 IF C=Z THEN GO TO BU
9940 LET A=A+I: PRINT C$:CHR$ PE
EK A: PRINT : LET C=C+J: GO TO B
F
9942 PRINT #H,A$: PAUSE H
9943 IF INKEY$=""c" THEN GO TO B
W
9944 IF INKEY$=""p" THEN COPY :
GO TO BW
9945 GO TO BV
9946 CLS : LET D=H: LET C=D: LET
X=AH: LET Y=AI: IF PEEK A=AF TH
EN GO TO BT
9948 IF L>=BD THEN PRINT #H,B$:
PAUSE H: STOP
9949 GO TO BK
9951 LET A=A+I: LET B=PEEK A: IF
B=AE THEN RETURN
9952 GO TO BY
```

RENUMBER AND PRELIMINARY FLOWCHART PROGRAM

```

1 CLS
10 IF 5>1 THEN GO TO 9000
20 FOR f=1 TO 3: PRINT 5: NEXT
f
30 GO SUB 400
40 LOAD ""
50 PRINT 1
60 STOP
70 LOAD "a"
80 SAVE "a" LINE 10: BEEP .3,1
2: BEEP 1,12: VERIFY "a": BEEP .
3,16: BEEP 1,16
90 NEW
100 LIST 10
350 STOP
400 PRINT 123: RETURN
410 STOP
:000 LET A=PEEK BZ+BB*PEEK CA: L
ET C=H: LET Y=A1: LET X=AH
:001 LET L=PEEK (A+1)+BB*PEEK A:
LET A=A+M: IF L>=BD THEN GO TO
BX
:002 PRINT L;TAB 0;CHR$ PEEK A:
LET C=C+J: PRINT
:003 PLOT X,Y: LET B=PEEK A: IF
B=AK OR B=AM OR B=AP OR B=AU THE
N GO TO BL
:004 IF B>AL AND B<AN OR B=AO OR
B=AQ OR B=AT OR B=AW OR B=AX OR
B=AZ THEN GO TO BM
:005 IF B=AV OR B=AY THEN GO TO
BN
:006 IF B=AR OR B=AS OR B=BA THE
N GO TO BQ
:007 DRAW V,H: DRAW N,Q: DRAW AC
,H: DRAW M,R: DRAW S,H
:008 LET Y=Y-R: IF B=AK OR B=AR
OR B=AM OR B=BA THEN GO TO BH
:009 PLOT X,Y: DRAW H,P: DRAW M,
M: DRAW N,N: DRAW N,M
:010 LET Y=Y-R
:011 LET B=PEEK A: IF B=AE THEN
GO SUB BY
:012 IF B=U THEN LET A=A+M
:013 IF B=T THEN GO TO BJ
:014 IF B=AF THEN GO TO BS
:015 LET A=A+1: GO TO B1
:016 IF C=Z THEN GO TO BU
:017 LET A=A+1: GO TO BE
:018 DRAW S,H: DRAW M,Q,E: DRAW
AA,H: DRAW N,R,E: DRAW S,H: GO T
O BG
:019 DRAW V,H: DRAW H,Q: DRAW AD
,H: DRAW H,R: DRAW V,H: GO TO BG
:020 DRAW R,N: DRAW R,H: DRAW N,
M: DRAW M,N: DRAW N,N: DRAW M,M:
DRAW Q,H: DRAW Q,N: DRAW Q,M: D
RAW R,M: IF B=AV THEN GO TO BP
:021 LET A=A+1: LET F=PEEK A: IF
F=U THEN LET A=A+M
:022 IF F<>AJ THEN GO TO BO
:023 LET A=A+1: LET F=PEEK A: PR
INT AT C-J,W;CHR$ F: PRINT : IF
F=AR OR F=AS THEN GO TO BR
:024 GO TO BG
:025 CIRCLE X,Y-M,M: PLOT X+M,Y-
M: DRAW AG,H: DRAW N,M: DRAW M,N
: DRAW N,N: PLOT X,Y: IF B=BA TH
EN GO TO BG
:026 LET A=A+1: LET F=PEEK A
:027 IF F<>U THEN GO TO BR
:028 LET A=A+K: LET G=PEEK A+BB*
PEEK (A+1): PRINT AT C-J,AB;G: P
RINT : GO TO BG
:029 IF C=Z THEN GO TO BU
:030 LET A=A+1: PRINT C$;CHR$ PE
EK A: PRINT : LET C=C+J: GO TO B
F

```

```

:031 PRINT #H;A$: PAUSE H
:032 IF INKEY$="c" THEN GO TO B
W
:033 IF INKEY$="p" THEN COPY :
GO TO BW
:034 GO TO BV
:035 CLS : LET D=H: LET C=D: LET
X=AH: LET Y=A1: IF PEEK A=AF TH
EN GO TO BT
:036 IF L>=BD THEN PRINT #H;B$:
PAUSE H: STOP
:037 GO TO BK
:038 LET A=A+1: LET B=PEEK A: IF
B=AE THEN RETURN
:039 GO TO BY

```

MERGED Demo and Final Version
of FLOWCHART PROGRAM

NOTE: use CLS and GO TO 10000
to operate this program

TYD☆BYTS

DON'T CLOG YOUR MEMORY WITH GOSUBs

Perhaps you didn't know that you can exhaust the memory of your TS 2068 in a surprisingly short time, if you repeatedly use a GOSUB without a corresponding RETURN.

Try the following:

```

100 PRINT "FREE BYTES: ";FREE
110 LET N=1
120 PRINT AT 3,0;N
130 LET N=N+1
140 GOSUB 120

```

and RUN the program. You'll get an OUT OF MEMORY report when N=78.

What happens, of course, is that the the computer saves up all of those addresses that it should have RETURNed to, stacking them neatly in a corner of the memory that it has reserved for this purpose. When that corner is full, you're OUT OF MEMORY, regardless of how many empty bytes may be lying around elsewhere.

You can get a bit more of an insight by adding the following line:

```
10 DIM A(1000)
```

You'll see that, although FREE (the number of free bytes of memory) has been decreased by 5000, the number of unrequited GOSUBs that the memory can accommodate is unchanged.

You might liken this effect to an embolus that blocks a strategic blood vessel, shutting down the body even though the rest of the cardio-vascular system is unimpaired.

THE MORAL

Obviously, the first thing to make sure of is to provide a RETURN for every GOSUB you include in the program. But a more subtle point--make sure that

the RETURN command ACTUALLY GETS USED. There are two very real possible pitfalls:

First, watch the program itself. It's all too easy to route the program around that vital RETURN (as the result, for instance, of an IF . . . THEN statement).

Second, be careful not to BREAK the program in the middle of a subroutine. If you have a program that you know you will want to BREAK repeatedly, include a command that will BREAK the operation at some controlled point. Something along the line of

```
IF INKEY$=B THEN GO TO . . .
```

followed by some address that will stop the program without interrupting a subroutine.

Of course, you can also sweep the debris out of that corner of memory by using the RUN or CLEAR instruction, but only at the cost of losing every other bit of data that the computer has saved up for you.

I suppose that the same principle applies to the ZX81/TS1000 as well, but the limit seems to be well above the practical range. I ran out of patience (with the TS1500, actually--I hope that I'm right in assuming that the same rule holds for the 1000) when N got up to 1,200.



Earl Dunnington
TS2068

Mr. Dunnington tells us that this quiz was taken from the Readers Digest for July 1988. In the form that he sent it, the questions and answers were quoted verbatim; with true editorial timidity, I changed the wording rather drastically, in order to minimize the danger of copyright violations. The principle remains intact.

The quiz itself goes in a pretty straightforward manner: you get the question, and choose an answer. The computer then tells you whether you have the right answer, to the accompaniment of some rather clever SOUND (Line 2010) for a wrong answer, or a happy series of BEEPs (Line 2030) for the correct one.

The program also keeps track of correct responses, and gives a percentage grade at the end of the run.

The first thing that catches the eye is the rather haphazard routing of program (see the "road map" for a directory of line numbers). I had to look at the DIGEST article before I realized what had happened: Earl had followed faithfully the progress of the printed quiz, which had scrambled the order of questions and answers in order to discourage peeking, whether voluntary or otherwise. Which is a pretty efficient programming technique, even though it does confuse the poor guy/gal who tries to analyze the program.

Earl also wanted it pointed out for the sake of Roger Phelps of Ithaca, MI that lines 2000/2005 and 2020 give us another look at two ways to print in the lower portion of the 2068 screen.

There's one curious inconsistency in the program, though. On the one hand, the writer is unnecessarily frugal in some aspects of memory use, such as using VAL "1020" for 1020 (unnecessarily because the program doesn't begin to exhaust the 2068--but then, why not be frugal for the sake of

NATIONAL LANDMARK QUIZ

frugality itself, as my Vermont-farmer ancestors would have said). At the same time, the lines with numbers ending in 1, 2, or 3 are repeated time and again, with slight changes. These could logically have been tucked away into subroutines.

	ques	ans	wrng	wrng
			ans	ans
Golden Gate	1010	1060	1100	
Yellowstone	1080	1110	1030	
Rushmore	1130	1160	1040	1090
Hawaii	1020	1070	1120	
Trees	1050	1200	1140	
Martin Luther King	1150	1210	1190	1240
Statue of Liberty	1180	1170	1220	1260
Tallest building	1270	1290	1230	1250

"Road map" for LANDMARKS program

And this all suggests an idea for those of you that have classroom applications. Why not a "generic" quiz framework, with the questions (and answers) stored in an array, so that you can use any one of several pre-set quizzes, either by LOADING the appropriate array or by calling up the appropriate one of several arrays that are stored in the computer? You could even make a sort of LOADER program (similar to those used in generating machine code sequences) to help you to generate the arrays with a minimum of repetitious routine work.

```

1 REM National Landmarks
programed by Earl Dunnington
2 REM FOR EDUCATIONAL USE
ONLY
3 REM This test is based on
the article How Well Do You Know
Your National Landmarks by Ken
Levine that appeared in the July
1988 Readers Digest
10 BORDER VAL "4": PAPER VAL "
6": CLS : PRINT AT SGN PI, VAL "4
"; "FOR EDUCATIONAL USE ONLY"; AT
VAL "7", VAL "6"; "NATIONAL LANDMA
RKS"; AT VAL "9", SGN PI; "programm
ed by Earl Dunnington"
20 PRINT AT VAL "16", SIN PI; "T
his test is based on the arti-c
le "How Well Do You Know Your
National Landmarks" by Ken Levin
ethat appeared in the July 198
8Reader's Digest ( 1988 The R
eader's Digest Association, Inc)"
: GO SUB VAL "2020"
30 LET SCORE=VAL "12"
1010 CLS : PRINT "FIRST, THE GOL
DEN GATE BRIDGE CONNECTS SAN F
RANCISCO WITH""(A) OAKLAND""
(B) SAUSALITO": GO SUB VAL "2000
"
1011 IF A$="A" OR A$="a" THEN G
O TO VAL "1100"
1012 IF A$="B" OR A$="b" THEN G
O TO VAL "1060"
1013 CLS : GO TO VAL "1010"
1020 CLS : PRINT "REMEMBER PEARL
HARBOR, SITE OF THE ATTACK THA
T BROUGHT AMERICA INTO WORLD WAR
II? PEARL HARBOR IS LOCATED AT
WHICH OF THE HAWAIIAN ISLAN
DS?""(A) OAHU""(B) HONOLULU"
: GO SUB VAL "2000"
1021 IF A$="A" OR A$="a" THEN G
O TO VAL "1070"
1022 IF A$="B" OR A$="b" THEN G
O TO VAL "1120"
1023 CLS : GO TO VAL "1020"
1030 GO SUB VAL "2010": PRINT "S
ORRY. OLD FAITHFUL IS KNOWN FOR I
TS PUNCTUALITY, RATHER THAN ITS
EIGHT. ALTHOUGH THE CORRECT A
NSWER IS NOW OBVIOUS, LET'S GO T
HROUGH THE QUESTION AGAIN, TO F
IX IT IN YOUR MIND.": GO SUB VAL
"2020": GO TO VAL "1080"
1040 GO SUB VAL "2010": PRINT "N
O. THOMAS JEFFERSON, OUR THIRDP
RESIDENT AND AUTHOR OF THE D
ECLARATION OF INDEPENDENCE, IS T
HERE.""HINT: WHICH ONE OF THE
HISTORIC-AL FIGURES NAMED WAS NO
T A PRESIDENT?": GO SUB VAL
"2020": GO TO VAL "1130"
1050 PRINT "WE ALL KNOW ABOUT CA
LIFORNIA'S GIANT SEQUOIA TREES
THAT GROW TO MORE THAN 30 FEET
IN DIAMETER AND TO OVER 300 FEET
IN HEIGHT. BUT THERE IS ANOTHER
TREE THAT GROWS EVEN TALLER. I
T IS""(A) THE BRISTLECONE PINE
""(B) THE REDWOOD": GO SUB VAL
"2000"
1051 IF A$="A" OR A$="a" THEN G
O TO VAL "1140"
1052 IF A$="B" OR A$="b" THEN G
O TO VAL "1200"
1053 CLS : GO TO VAL "1050"
1060 GO SUB VAL "2030": PRINT "R
IGHT. OAKLAND IS CONNECTED WITHS

```

```

AN FRANCISCO BY THE BAY BRIDGE."
: GO SUB VAL "2020": GO TO VAL "
1080"
1070 GO SUB VAL "2030": PRINT "R
IGHT. PEARL HARBOR IS LOCATED O
N THE island OF OAHU, NEAR THE c
ity OF HONOLULU.": GO SUB VAL "2
020": GO TO VAL "1050"
1080 CLS : PRINT "THE GEYSER IN
YELLOWSTONE NATIONAL PARK
THAT HOLDS THE RECORD FOR HEI
GHT OF ERUPTION IS""(A) OLD FA
ITHFUL""(B) STEAMBOAT GEYSER":
GO SUB VAL "2000"
1081 IF A$="A" OR A$="a" THEN G
O TO VAL "1030"
1082 IF A$="B" OR A$="b" THEN G
O TO VAL "1110"
1083 CLS : GO TO VAL "1080"
1090 GO SUB VAL "2010": PRINT "N
O. TEDDY'S THERE, EVEN BIGGER T
HAN LIFE. TRY AGAIN.": GO SUB VA
L "2020": GO TO VAL "1130"
1100 GO SUB VAL "2010": PRINT "S
ORRY, YOU'RE OFF ON THE WRONG F
OOT. TRY AGAIN.": GO SUB VAL "20
20": GO TO VAL "1010"
1110 GO SUB VAL "2030": PRINT "Y
ES. ALTHOUGH OLD FAITHFUL IS M
ORE PUNCTUAL, STEAMBOAT GEYSER H
AS BEEN KNOWN TO SPOUT ALMOST 4
00 FEET, AS COMPARED WITH 180 F
EET FOR OLD FAITHFUL.": GO SUB V
AL "2020": GO TO VAL "1130"
1120 GO SUB VAL "2010": PRINT "S
ORRY. HONOLULU IS NOT AN ISLAND B
UT A city, CAPITAL OF HAWAII. T
RY AGAIN.": GO SUB VAL "2020": G
O TO VAL "1020"
1130 CLS : PRINT "THE CARVINGS O
F THE FOUR PRESI- DENTS AT MOUNT
RUSHMORE IN SOUTH DAKOTA INCLUDE
ALL except THE FOLLOWING""(
A) THOMAS JEFFERSON""(B) THEOD
ORE ROOSEVELT""(C) BENJAMIN FR
ANKLIN": GO SUB VAL "2005"
1131 IF A$="A" OR A$="a" THEN G
O TO VAL "1040"
1132 IF A$="B" OR A$="b" THEN G
O TO VAL "1090"
1133 IF A$="C" OR A$="c" THEN G
O TO VAL "1160"
1134 CLS : GO TO VAL "1130"
1140 GO SUB VAL "2010": PRINT "N
O, SORRY. BRISTLECONE PINES AREA
MONG THE WORLD'S OLDEST TREES B
UT THEY'RE NOT THE TALLEST. TRY A
GAIN.": GO SUB VAL "2020": GO TO
VAL "1050"
1150 CLS : PRINT "WHERE DID MAR
TIN LUTHER KING JR'S AUGUST 19
63 MARCH FOR RACIAL EQUALIT
Y FINISH?""(A) THE WASHINGTON
MONUMENT""(B) THE LINCOLN MEMO
RIAL""(C) ARLINGTON NATIONAL C
EMETERY": GO SUB VAL "2005"
1151 IF A$="A" OR A$="a" THEN G
O TO VAL "1190"
1152 IF A$="B" OR A$="b" THEN G
O TO VAL "1210"
1153 IF A$="C" OR A$="c" THEN G
O TO VAL "1240"
1154 CLS : GO TO VAL "1150"
1160 GO SUB VAL "2030": PRINT "C
ORRECT. THE FOUR PRESIDENTS H
ONORED ON MOUNT RUSHMORE ARE G
EORGE WASHINGTON, THOMAS J
EFFERSON, ABRAHAM LINCOLN, AND T

```

```

EDDY ROOSEVELT."'"BENJAMIN FRAN
KLIN, THOUGH A TRE-MENDOUSLY INF
LUENTIAL FIGURE IN OUR COUNTRY'S
HISTORY, WAS NEVER ELECTED PRESI
DENT.": GO SUB VAL "2020": GO TO
VAL "1020"
1170 GO SUB VAL "2030": PRINT "R
IGHT--NOT ON ELLIS ISLAND, W
HIGH HAS HISTORICALLY BEEN A
SSOCIATED WITH IMMIGRANTS TO O
UR SHORES.": GO SUB VAL "2020":
GO TO VAL "1270"
1180 CLS : PRINT "IN 1886, THE F
RENCH GOVERNMENT GAVE US THE S
TATUE OF LIBERTY, WHICH HAS SINC
E BEEN A WELCOME SIGHT TO ALL T
RAVELERS ARRIVING AT NEW YORK. T
HE STATUE IS SITU-ATED ON'"(A)
LIBERTY ISLAND'"(B) ELLIS ISL
AND'"(C) CONEY ISLAND": GO SUB
VAL "2005"
1181 IF A$="A" OR A$="a" THEN G
O TO VAL "1170"
1182 IF A$="B" OR A$="b" THEN G
O TO VAL "1220"
1183 IF A$="C" OR A$="c" THEN G
O TO VAL "1260"
1184 CLS : GO TO VAL "1180"
1190 GO SUB VAL "2010": PRINT "S
ORRY. TRY AGAIN": GO SUB VAL "20
20": GO TO VAL "1150"
1200 GO SUB VAL "2030": PRINT "R
IGHT. GIANT REDWOODS GROW TO O
VER 350 FEET. ALTHOUGH AMONG T
HE WORLD'S OLDEST (OVER 4000 Y
EARS OLD), BRISTLECONE PINES A
RE NOT AS TALL.": GO SUB VAL "20
20": GO TO VAL "1150"
1210 GO SUB VAL "2030": PRINT "C
ORRECT. LINCOLN WAS ONE OF OUR G
REAT HUMAN-RIGHTS ACTIVISTS, A
ND HIS MEMORIAL HAS BEEN THE S
ITE OF MANY DEMONSTRATIONS FOR R
ACIAL EQUALITY.": GO SUB VAL "20
20": GO TO VAL "1180"
1220 GO SUB VAL "2010": PRINT "N
OT QUITE. ELLIS ISLAND HAS BEEN T
HE POINT OF ENTRY FOR MILLIONS O
F AMERICA'S IMMIGRANTS, BUT T
HE STATUE OF LIBERTY IS LOCATED S
OMEWHERE ELSE. TRY AGAIN.": GO S
UB VAL "2020": GO TO VAL "1180"
1230 GO SUB VAL "2010": PRINT "N
OT QUITE. THE WORLD TRADE CEN- T
ER IS 1350 FEET TALL, THE NEXT-T
O-THE HIGHEST BUILDING IN THE W
ORLD. HAVE ANOTHER TRY.": GO S
UB VAL "2020": GO TO VAL "1270"
1240 GO SUB VAL "2010": PRINT "S
ORRY. TRY AGAIN": GO SUB VAL "20
20": GO TO VAL "1150"
1250 GO SUB VAL "2010": PRINT "I
T USED TO BE THE TALLEST, BUT N
OW THE EMPIRE STATE BUILDING IST
HIRD ON THE LIST. TRY AGAIN.": G
O SUB VAL "2020": GO TO VAL "127
0"
1260 GO SUB VAL "2010": PRINT "O
H, COME ON! TRY AGAIN.": GO SUB
VAL "2020": GO TO VAL "1180"
1270 CLS : PRINT "WHICH OF AMER
ICA'S SKYSCRAPERS IS THE WORLD'S
TALLEST INHABITED MAN-MADE STRUC
TURE, NOT COUNTING ANTENNAS OR OT
HER ROOF OBJECTS?"'"(A) THE WOR
LD TRADE CENTER'"(B) THE EMPIR
E STATE BUILDING'"(C) THE SEAR
S TOWER IN CHICAGO": GO SUB VAL

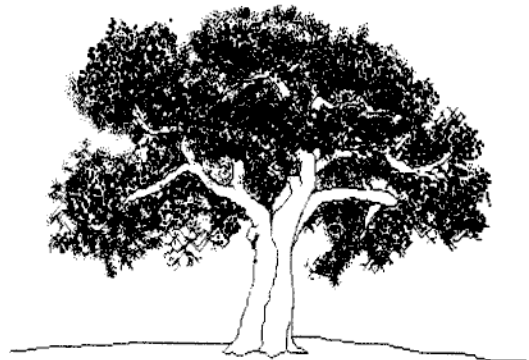
```

```

"2005"
1271 IF A$="A" OR A$="a" THEN G
O TO VAL "1230"
1272 IF A$="B" OR A$="b" THEN G
O TO VAL "1250"
1273 IF A$="C" OR A$="c" THEN G
O TO VAL "1290"
1274 CLS : GO TO VAL "1270"
1280 PRINT '"A-PLAY AGAIN'"'"B-
RETURN TO BASIC'"'"C-QUIT": GO S
UB VAL "2005"
1281 IF A$="A" OR A$="a" THEN C
LS : GO TO VAL "20"
1282 IF A$="B" OR A$="b" THEN S
TOP
1283 IF A$="C" OR A$="c" THEN S
TOP
1284 CLS : GO TO VAL "1280"
1290 GO SUB VAL "2030": PRINT "T
HAT'S IT. THE SEARS TOWER IS 1
454 FEET HIGH, THE WORLD TRADE C
ENTER 1350, AND THE EMPIRE S
TATE BUILDING ONLY 1250. '" TO
CONTINUE PRESS ANY KEY": PAUSE S
IN PI
1300 CLS : LET PERCENT=VAL "INT
(SCORE/12*100)": PRINT "CONGRATU
LATIONS. YOU'RE DONE'"YOUR
SCORE WAS: ";PERCENT;"%": GO TO
VAL "1280"
2000 INPUT "PRESS<A>OR<B>THEN<EN
TER>";A$: RETURN
2005 INPUT "PRESS<A>,<B>,OR<C>TH
EN<ENTER>";A$: RETURN
2010 BORDER VAL "2": CLS : }8,16:
}12,200;13,0;}7,62: FOR P=SGN PI
TO VAL "100": }0,P: PAUSE VAL "2
": NEXT P: }8,16;9,16;10,16;}6,25
;}13,0;}12,70;}7,7: PAUSE VAL "1
00": LET SCORE=SCORE-VAL "1": RE
TURN
2020 PRINT #SGN PI;"PRESS <ENTER
> TO CONTINUE": PAUSE NOT PI: BO
RDER VAL "4": CLS : RETURN
2030 CLS : BEEP .25,24: BEEP .25
,26: BEEP .25,28: BEEP .5,31: BE
EP .25,28: BEEP 1,31: RETURN
9900 REM SAVE
9950 SAVE "NL" LINE 1
9960 BEEP .3,12: BEEP 1,12
9970 CLS : PRINT '"TO VERIFY,
RE-PLAY TAPE"
9980 VERIFY ""
9990 BEEP .3,20: BEEP 1,20

```

LISTING for LANDMARKS program



TYD ☆ BYTS

Take Care of Those 2040 Printouts

As we all have discovered, to our sorrow, those printouts produced by the TS2040 printer are pretty fragile stuff. No paper has an unlimited life, of course, as witness the incredible lengths to which libraries go to try to protect their most valuable books. But your printer's paper reminds one of the poet who wanted his epitaph to read "Here lies one whose name is writ in water".

To begin with, the paper is temperature-sensitive. That's how it works. So it goes without saying that exposure to heat or sunlight is going to impair its usability. For instance, I bought some rolls of the double-width Big Blue paper, with the intention of splitting them to fit the 2040 printer. A great bargain, but you should have seen what happened to the paper when I split it with my bandsaw. Luckily, that big black smear was out at the edge, where it doesn't interfere with the printing. Paul Holmgren tells me that he got a noticeable discoloration just from using a hacksaw.

Another problem with the paper is that it seems to have a special absorbency for glue. I tried mounting some printouts on stiff paper, and found after a few days that the glue had made serious stains. Similarly when I tried splicing two printouts together with transparent tape. Even the so-called PERMANENT tape soon makes the printing illegible—even when it's applied to the back of the thermal paper.

The only solution that I have found is to photocopy the printouts IMMEDIATELY. Well, within a couple of days, anyway. Then your records will have the permanence of the photocopy paper, whatever that happens to be. You can be sure that it'll be many orders of magnitude better than that of the thermal paper.

Or you could SAVE the program on tape or disk, of course, and generate a new printout whenever you want one.

One technique I got from Bob Swoger--the cores of bathroom tissue rolls just fit the 4-1/2 inch paper, and make great mailing tubes. Far better than folding the printout, and hoping that you don't damage it in the process.

While we're at it, Frank Davis tells me that Radio Shack makes a thermal paper that almost fits the 2040. The trouble is that it's a fraction of an inch narrower, and tends to wander. Frank invested 50 cents (two quarters) and inserted them (standing on edge) in the printer's paper well, one on each side of the paper roll. Workers with more sturdy materials call this sort of wedge a "shim".

Of course, you can use Susan B. Anthony dollars instead of quarters, if you want to be ostentatious
...

A Loser on a Loser

Serious bridge players recognize the value of the tactic of playing a loser on a loser. Suppose, for instance, that you (declarer) have a small diamond in your hand, and a small heart in dummy. Spades are trumps. At best, you'd like to take a trick with each of these small cards, either by trumping them or by leading them after your opponents have nothing left but clubs. But you can't always manage things so neatly. The next best tactic, if you can arrange it, is to get rid of them both on the same trick, rather than losing a separate trick with each.

Similarly in programming. If you have two operations that take lots of time, see if you can arrange the program so that you can combine the delays of both operations, with a saving in overall delay. Or at least a saving in the frustration generated by the delay.

A principal cause for delay in computer operation is the response time of the human senses and muscles. Here's a way that you can "bury" an unavoidable delay into that response time.

The program shown is a simple sort routine. The trick is that it sorts each number WHEN IT IS ENTERED, so the time of sorting tends to get lost in the time it takes to INPUT the number. Even in BASIC, which is not known for its high speed, the sort time is not detectable, unless you bury the new number way down below hundreds of numbers already in the array. If you used machine code, even that delay would be impossible to detect.

This technique also accommodates transcendental numbers, by the way, as well as fractions, negative numbers, and numbers expressed in engineering notation, such as 2E4.

```
10 LET a=1000
20 DIM r(a)
30 LET count=0
110 INPUT n$
120 IF n$=" STOP " THEN GO TO 1
000
130 LET no=VAL n$
140 FOR f=count TO 1 STEP -1
150 IF no>=n(f) THEN GO TO 180
160 LET n(f+1)=n(f)
170 NEXT f
180 LET n(f+1)=no
190 LET count=count+1
200 IF count<a THEN GO TO 100
1000 FOR f=1 TO count
1010 PRINT n(f); " ";
1020 NEXT f
```

Low-delay SORTING program

CLASSIFIED

4 SALE:\$350 TS2068 W/AERCO INTER-
FACE, DUAL WAFADRIVE, 2 SPECTRUM
OP SYSTEMS, LIGHT PEN, SPECTRUM
JOY STICK, 2 WORD PROCESSORS, &
100'S OF OTHER PROGRAMS OVER 100
CASSETTES, 14 WAFERS 10-128K &
TECH DATA. G.LIPSCOMB, 3402
BEAUMONT RD, DALE CITY, VA 22193

SURGE PROTECTORS. PROTECT YOUR
COMPUTER AND PERIPHERALS FROM
DAMAGE DUE TO VOLTAGE SURGES
OR SPIKES. PRICED FROM \$34.95.
WRITE FOR PRICE LIST, DESCRIB-
ING YOUR NEEDS, TO: KEY
ASSOCIATES, PO BOX 5735,
CLEARWATER, FL 34618-5735.

Send ad material along with
payment to:

CLASSIFIED AD DEPT.
SyncWare News
1413 Elliston Drive
Bloomington, IN 47401

LARKEN PRESENTS ...

UP TO 256K RAM for your 2068

- Expand your 2068 with up to 256K of battery backed up Ram
- Larken Operating system lets you SAVE to memory, just like cassette or disk. (Floppy disk not required)
- All Cassette commands supported. Very Fast and Reliable.
- Can be used with ALL existing 2068 or Spectrum software.
- Uses the new 32K static ram chips, 62256LP or 43256LP
- System consists of Larken Cartridge and Rear Memory Board.
- ** PRICE - MEMORY SYSTEM with 64K Ram \$129.00
- MEMORY SYSTEM with 0 K \$ 95.00

LARKEN 2068 FLOPPY DISK SYSTEM

- The most advanced Dos available for the 2068/Spectrum. LKdos uses ALL Commands such as CAT MERGE ERASE LOAD SAVE PRINT OPEN etc. Also can support RAMDISK up to 256K and Sequential / Random Access Files (with additional software). The Larken Disk Interface can handle up to 4 floppys for up to 3.2 Megabytes of storage. Also NMI Snapshot Save Button and KEMPSTON Joystick port on interface Also 10 Extended Basic commands for Windows and Graphics.

AERCO RAMEX or OLIGER disk users can add LKdos for more commands, Ramdisk and access to all LKdos software

- ** PRICE - Larken Floppy Disk System \$119.95
 - Floppy Disk IF with 0 K Memory board .. \$169.95
 - Larken Disk Editor \$ 15.00
 - Sequential/Random access files \$ 15.00
 - Xmodem to Disk Modem package \$ 15.00
 - ZX-81 Floppy Interface (15 left) ... \$ 99.95
 - LKDOS for Aerco,Ramex or Oliger Disk IF \$ 59.95
- (All prices are US , Add 6% Shipping)

LARKEN ELECTRONICS RR#2 NAVAN ONTARIO CANADA K4B-1H9
(613)-835-2680

SyncWare News

CONTENTS

For Your Support.....	2
Forum.....	3
Tyd*Byts.....	5
NVM and the 2068 (Holmgren).....	6
ShareWare -- The T/S Lifeline? (Nachbaur).....	7
Getting Started With Beta BASIC: Part II (Hartung).....	9
Basil's Compendium (Wentworth)....	12
THE FLOWCHART CHALLENGE: -- The "Old Gent's" Flowcharter (The Old Gent).....	15
-- The Dunnington Flowcharter (Dunnington).....	17
Tyd*Byts.....	19
National Landmark Quiz (Dunnington).....	20
Tyd*Byts.....	23

SyncWare News

602 S. Mill St., Louisville, OH 44641

ADDRESS CORRECTION REQUESTED

Bulk Rate
U.S. POSTAGE
PAID
Louisville, OH
Permit No. 40

Editor:
Basil Wentworth
1413 Elliston Dr.
Bloomington, IN 47401

Technical Editor:
Fred Nachbaur
C-12, Mtn. Stn. Group Box
Nelson, BC V1L 5P1 Canada

Subscriptions, Advertising:
Jeffrey Moore
602 S. Mill Street
Louisville, OH 44641

Submissions, announcements, letters to the Editor,
and all other material of editorial interest should
be sent to the Indiana address.